سلسلة ملخصات شوم

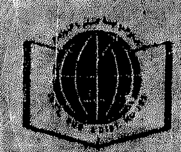
نظريات ومسائل فى

# البرمجة بلغة الكوبل

يتضمن الكوبل الهيكلى

لورنس نيوكومر

يحتوى الكتاب على ٦٨٤ مسألة محلولة



الدار الدولية للنشر والتوزيع



# ملخصات شوم
## نظريات ومسائل
## فى
# البرمجة بلغة الكوبل


تاليف
**لورنس نيوكومر**
أستاذ علوم الحاسب المساعد
جامعة ولاية بنسلفانيا

ترجمة ومراجعة
**د. مهندس/ سرور على إبراهيم سرور**
كلية الاقتصاد والإدارة
جامعة الملك سعود - فرع القصيم





**الدار الدولية للنشر والتوزيع**
**القاهرة - الكويت - لندن**

# حقــوق النشــر

• الطبعة الأنجليزية : حقوق التأليف © ١٩٨٤ دار ماكجروهيل للنشر، إنك، جميع الحقوق محفوظة.

Schaum's Outline of theory & Problems of Programming with Structured Cobol

by

LAWRENCE R. NEWCOMER

• الطبعة العربية الأولى : حقوق الطبع والنشر © ١٩٩١، جميع الحقوق محفوظة للناشر

**الدار الدولية للنشر والتوزيع**

ص . ب ٥٥٩٩ هليوبوليس غرب – القاهرة

ت : ٢٥٨٢٨٨٧

تلكس : ٢٠٠٧٠ PBCRB UN

فاكس : ٢٩١٨٠٥٩ / ٢٠٢ ٠٠



رقم الإيداع ١٩٩١/٣٤٥٨

I.S.B.N. 977-5107-26-1

بِسْمِ اللهِ الرَّحْمَنِ الرَّحِيمِ

« قُلْ لَوْ كَانَ ٱلبَحْرُ

مِداداً لِكَلِمَاتِ رَبِّي

لَنَفِدَ ٱلبَحْرُ قَبْلَ

أَنْ تَنْفَدَ كَلِماتُ رَبِّى،

وَلَوْ جِئْنَا بِمِثْلِهِ

مَدَداً »

صَدَقَ اللهُ العَظِيمُ

# مقدمة المؤلف

هذا الملخص يعلم الكوبل ( كما عرفه المعهد القومى الامريكى للنمطيات عام 1974 ) * من وجهة نظر البرمجة المرتبة structured Programming** فهو يغطى معظم ، وليس كل ، الاساليب التى تمكن المبرمجين من زيادة انتاجيتهم فى اعداد برامج فردية . وتركز على استخدام تصميم الاجزاء من القمة الى القاعدة، مع تطبيق نمطيات متعددة من شفرة الكوبل .

ويهدف الكتاب إلى ان يكون شاملاً وموجزاً ، سامحا للقارىء بالتقدم السريع . ومع نهاية الفصل الثانى يكون القارىء معداً لكتابة برامج بسيطة بلغة الكوبل . وحيث إن تعلم برمجة الكمبيوتر يشبه تعلم السباحة ( عليك بالقفز واداء العمل ) فمطلوب من القارىء أن ينفذ أكبر قدر ممكن من تمارين البرمجة مستخدماً جهاز كمبيوتر فعلى .

وقد صمم الفصل التاسع ، الذى يغطى التصحيح ؛ بحيث يمكن قراءته فى أى وقت بعد اتمام قراءة الفصل الثانى، وعليك أن ترجع إليه كلما وجدت مشكلة فى تمارين البرمجة .

عند ظهور امتدادات لكوبل 1974 النمطى ( القياسى) فهى امتداد IBM OS/VS ، وسوف تعرف بهذا الاسم بوضوح*** . وكل الأمثلة ما هى إلا مشاكل سبق اختبارها فى جامعة ولاية بنسلفانيا الامريكية باستخدام نظام كمبيوتر IBM 3081 وكوبل IBM OS/VS . وحيث إن المعلومات تختلف من نظام لآخر.. فقد قدمنا مادة مرتبطة تماماً بنظم من طراز IBM 370 ، مع نظم تشغيل من نوع OS/VE . ويعمل المعهد القومى الأمريكى للنمطيات حاليا على صيغة 1980 للكوبل القياسى . والتغييرات المحتمل إدخالها فى كوبل الثمانينيات القياسى مقدمة فى ملحق جـ **** .

المؤلف

---

* لقد صدر عن نفس المعهد صيغة مطورة من لغة الكوبل عام ١٩٨٥، وسوف يتم التعرض لبعض السمات التى تميز هذه الصيغة عن الصيغة المعروفة عام ١٩٧٤ فى ملحق جـ ( المترجم )

** يطلق البعض اسم البرمجة الهيكلية أو البرمجة المتسلسلة على نفس الاسم الا أن المترجم يفضل اسم البرمجة المرتبة لما له من وقع افضل لدى القارىء ( المترجم )

*** يجب التنويه أن هناك عديداً من صيغ الكوبل الاخرى المتاحة تجاريا ومنها MS-COBOL

**** لقد ظهرت فعلا صيغة الثمانينات وهى كوبل ١٩٨٥ النمطى وسوف يشتمل ملحق C على التغييرات الفعلية لهذه الصيغة عن صيغة الكوبل ١٩٧٤ (المترجم)

# مقدمة الناشر

المعرفة هى أصل الحضارة

والكلمة هى أصل المعرفة،

والكلمة المطبوعة هى أهم مكون فى هذا المصدر.

وقد كانت الكلمة المطبوعة ولاتزال أهم وسائل الثقافة والأعلام وأوسعها إنتشاراً وأبقاها أثراً، حيث حملت إلينا حضارات الأمم عبر السنين لتتولى الأجيال المتلاحقة صياغة حضارتها وإضاءة الطريق بنور العلم والمعرفة.

والكلمة تبقى مجرد فكرة لدى صاحبها حتى تتاح لها فرصة نشرها وترجمتها إلى لغات الآخرين، ثم توزيعها، وذلك وحده هو الذى يكفل لها أداء رسالتها.

وعالم الكتب العلمية عالم رحب ممتد الآفاق، متسع الجنبات، والعلم لا وطن له ولا حدود، ويوم يحظى القارىء العربى بأحدث الكتب العلمية باللغة العربية لهو اليوم الذى تتطلع له الأمة العربية جمعاء.

والدار الدولية للنشر والتوزيع تشعر بالرضا عن مساهمتها فى هذا المجال بتقديم الطبعات العربية للكتب العلمية مستهدفة توفير احتياجات القارىء العربى أستاذاً وباحثاً وممارساً.

والله ولى التوفيق،،

محمد وفائى كامل

| الصفحة | المحتويات | مسلسل |
|---|---|---|

| مسلسل | المحتويات | الصفحة |
|---|---|---|

| مسلسل | المحتويات | الصفحة |
|---|---|---|

مسلسل | المحتويات | الصفحة

| مسلسل | المحتويات | الصفحة |
|---|---|---|

| مسلسل | المحتويات | الصفحة |
|---|---|---|

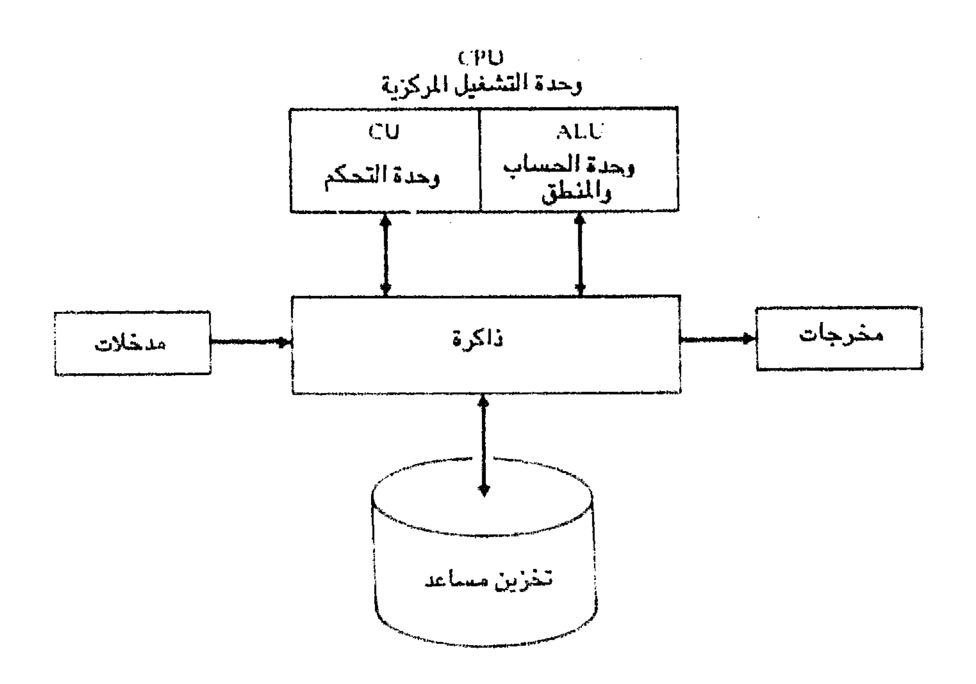
# الفصل الأول

# مفاهيم تشغيل البيانات

# Data Processing Concepts

## ١-١ كيف يعمل نظام الكمبيوتر

يشار الى الوحدات الطبيعية التى يتكون منها نظام الكمبيوتر بنظم المكونات hardware ، ومن وجهة نظر المبرمج .. توجد للكمبيوتر مكونات وظيفية مبينة فى شكل (١-١)




شكل (١ - ١)

## ١ - ٢ المدخـــــلات

وحدات المدخلات هى آلات تنقل معلومات من العالم الخارجى الى داخل ذاكرة الكمبيوتر . وفى معظم الاحوال .. علينا ان نضع المعلومات فى صورة مناسبة لاستخدامها بواسطة وحدة المدخلات أولا ، وعادة مايكون ذلك باحدى صور الطباعة . وتسمى المستندات التى تطبع منها المعلومات بمستندات المصدر Source documents . وحدات المدخلات الثلاث الرئيسية هى مايلى :

### قارئـــات بطاقـــات

يستخدم مثقب بطاقات card punch اولا فى تسجيل المعلومات كثقوب فى بطاقات لها حجم نمطى . وتوضع مجموعة البطاقات بعد ذلك فى قارىء بطاقات، يشعر - بطريقة كهربائية ميكانيكية او بطريقة ضوئية ميكانيكية - بالثقوب الموجودة فى البطاقات، وينقل المعلومات الى ذاكرة الكمبيوتر .

### من المفتاح إلى القرص

يشبه نظام من المفتاح الى القرص فى مفهومه نظام البطاقات ، إلا انه يسجل البيانات مغناطيسيا، على قرص مرن floppy disk او على قرص صلب hard disk. عند ذلك يوضع القرص فى مشغل اقراص disk drive ، يستطيع الاحساس بالمعلومات المخزنة مغناطيسيا وينقلها الى ذاكرة الكمبيوتر . يقرأ مشغل الاقراص البيانات بسرعة اسرع كثيرا من قارىء البطاقات . وعلى عكس البطاقات.. فمن الممكن اعادة استخدام الاقراص المرنة أو الاقراص الصلبة لبيانات مختلفة .

### نسخ دائمة ونهايات طرفية من أنبوب أشعة الكاثود

يمكن توصيل نهايات طرفية بالكمبيوتر بحيث تخدم لوحات مفاتيحها keyboards كوحدات مدخلات . وعند الضغط على احد المفاتيح.. ينقل الرمز لهذا المفتاح الى ذاكرة الكمبيوتر . وللنهايات الطرفية الذكية intelligent terminals مشغل دقيق microprocessor (كمبيوتر دقيق على رقيقة متكاملة )، وبعض الذاكرة المبنية داخل النهاية الطرفية؛ بحيث تكون النهاية الطرفية قادرة على تخزين الرموز عند كتابتها ونقلها إلى الكمبيوتر ، على هيئة مجموعات ( بدلاً من نقلها رمزاً رمزاً ) . كما يمكن ان تجرى النهايات الطرفية الذكية بعض التشغيل المحلى للمعلومات داخل النهاية الطرفية نفسها . وكل من النهايات الطرفية الذكية والغبية (الصماء) dumb لها وحدات مخرجات كذلك كجزء من النهاية الطرفية :

للنهايات الطرفية للنسخ الدائمة hard copy طابعات تشبه الآلات الكاتبة، بينما لايكون للنهايات الطرفية التى لها انبوب اشعة كاثود (CRT) cathode- ray-tube الا شاشة مرئية.

## ١ - ٣ المخـــــرجات

تستقبل وحدات المخرجات output المعلومات من ذاكرة الكمبيوتر، وتنقلها إلى العالم الخارجى سواء كانت فى صورة مقروءة للإنسان human-readable أم فى صورة مقروءة لآلة machine-readable .

### الطابعـــات

تسجل هذه الوحدات محتويات ذاكرة الكمبيوتر كرموز مطبوعة على ورق . وفى طابعات الطَّرق impact printers .. يضغط شريط الحبر على الورق، أما فى الطابعات التى لاتستخدم الطَّرق non impact printers .. فيمكن استخدام

وسائل حرارية او اليكتروستاتيكية او اشعة ليزر فى تكوين شكل الرمز على الورق . تطبع طابعات الرموز character printer رمزا واحدا بينما تطبع طابعات الاسطر line printers سطرا كاملا فى نفس الوقت كما تستطيع طابعات الاسطر التى تعمل بأشعة الليزر طباعة مايزيد عن 20000 سطر فى الدقيقة الواحدة .

## النهايات الطرفية للنسخ الدائمة، وأنابيب أشعة الكاثود

الوحدات التى سبق شرحها فى القسم ١ - ٢ كوحدات مدخلات ومخرجات معا .

## مثقبات بطاقات

تعمل هذه الوحدات تحت تحكم مباشر من وحدة التشغيل المركزية، فتثقب المعلومات على هيئة ثقوب فى البطاقات .

## القرص الصلب

تستطيع مشغلات الاقراص التى تدخل معلومات من اقراص مرنة أو اقراص صلبة ان تسجل كذلك معلومات على الاقراص.

## ١ - ٤ التخزين المساعد

هى وحدات قادرة على كل من المدخلات والمخرجات، ويمكن ان تخدم كوحدات تخزين مساعد auxiliary storage devices وهى مطلوبة بسبب :

( ١ ) حتى ذاكرات الكمبيوتر الكبيرة لايمكنها ان تحتفظ بكل المعلومات اللازمة لمجال اعمال صغيرة نسبيا .

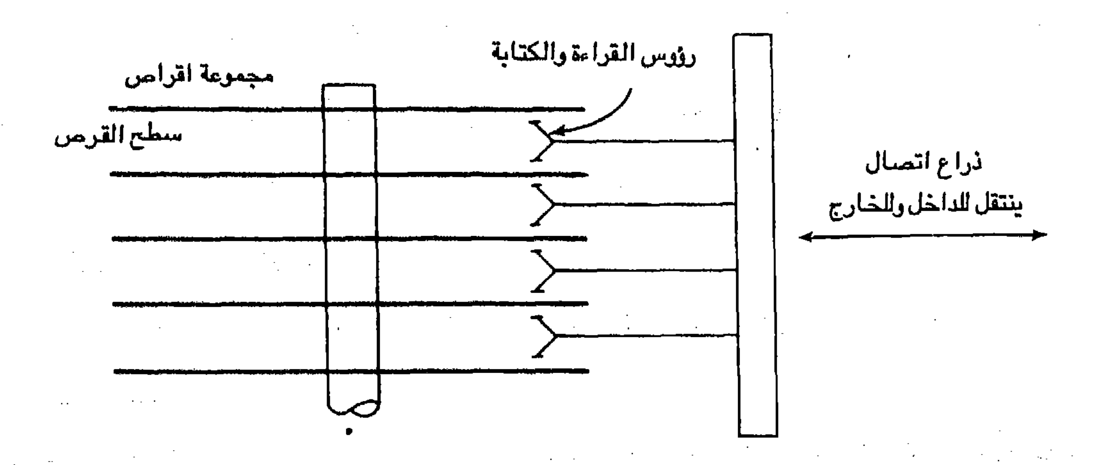
( ٢ ) عند انقطاع التيار عن اجهزة الكمبيوتر.. تمحى كل المعلومات الموجودة فى ذاكرة الكمبيوتر ، فاذا تواجدت كل المعلومات المطلوبة داخل الذاكرة ، فستكون هناك مخاطرة غير مقبولة بحفظها فى الذاكرة .

## القرص الصلب

فى الجزء الذى يعد اهم وحدة تخزين مساعد.. يستخدم مشغل الاقراص فى قراءة بيانات من مجموعة اقراص disk pack والكتابة عليها . وتحتوى مجموعة الاقراص على رصة من اقراص مسطحة من الالومنيوم مغطاة بمادة مغناطيسية موضوعة على محور مركزى ( شكل ١ - ٢ ) . يتحكم ذراع اتصال access arm فى مجموعة من رؤوس القراءة والكتابة read/write heads (رأس واحدة لكل سطح) ، ويمكنه ان ينتقل للداخل والخارج . فى موضع السكون.. يضع ذراع الاتصال الرؤس مواجهة لمجموعة من المسارات tracks الدائرية الموجودة على اسطح مجموعة الاقراص وتسجل البيانات على هذه المسارات . والمسارات مرتبة على هيئة مجموعات تسمى اسطوانات cylinders وهى مجموعات المسارات التى يمكن الاتصال بها فى اى موضع للسكون بآلية الاتصال .

يتراوح متوسط وقت نقل ذراع الاتصال الى الاسطوانة المطلوبة ( وقت الاتصال access time ) بين ١٦ و٦٠ (مللى ثانية millisecond هى واحد من الف من الثانية) . يجب ان يضاف وقت تأخر الدوران rotational delay الى هذا الوقت ( الوقت اللازم لوصول النقطة المطلوبة من المسار تحت الرأس ) والوقت ( الضئيل ) اللازم لنقل البيانات الى او من الذاكرة . ويكون اجمالى وقت الاسترجاع retrieval time حوالى ٣٠ مللى ثانية لاقراص IBM .

لبعض الاقراص ( تسمى ثابتة الرأس لكل مسار fixed head-per track) رأس لكل مسار من مسارات مجموعة الاقراص وبهذا يقل وقت الاتصال الى حوالى صفر .




شكل (١ - ٢)

## الشـــريط المغناطيســـى

يشبه الشريط المغناطيسى للكمبيوتر- فى الاحساس - شريط التسجيل المنزلى ويمكن لفه على بكرات او على هيئة كاسيت. تسجل المعلومات مغناطيسيا على طول الشريط.

وهناك عيب أساسى فى الشريط كتخزين مساعد، وهو انه يجب ان يقرأ مشغل الشرائط كل المعلومات التى تسبق قطعة المعلومات المطلوبة للحصول على هذه القطعة . وعلى هذا .. فالشريط كفء اذا ماأجرينا المعلومات فى نفس التتابع التى سبق تخزينها على الشريط به فقط ( تشغيل تتابعى sequential processing ) . يمكن تشغيل المعلومات المخزنة على القرص باى ترتيب نريده ( تشغيل عشوائى random processing ).

## ١- ٥ الذاكـــرة

تتكون ذاكرة الكمبيوتر من مكونات اليكترونية يمكن ان تخزن كل منها رقم ثنائى واحد ( binary digit ( bit من المعلومات وحيث ان (bit) يمكن ان يكون لها القيمة 0 أو 1 فقط .. فتجمع (bit) فى وحدات من ثمانية ( عادة ) تسمى بايت byte. والبايت الواحد هو جزء من ذاكرة الكمبيوتر يكفى لأن يخزن رمزا character واحداً من المعلومات .

وبايت الذاكرة مرقمة ٠ و ١ و ٢ و ٣ و ... والعدد الفردى المحدد لتعريف بايت يسمى عنوانا address . ويمكن ان يتحقق الاتصال العشوائى باى عنوان مطلوب فى حوالى نانو ثانية nano second (نانوثانية هى ١ من مليون من الثانية).

تقاس سعة الذاكرة وسعة التخزين المساعد بوحدات كيلوبايت kilobyte ( الكيلوبايت هو حوالى ١٠٠٠ بايت ) ووحدات ميجابايت megabyte (الميجابايت هو حوالى مليون بايت) . وتسمى الذاكرة التى سعتها حوالى ١٢٨٠٠٠ بايت ذاكرة لها ١٢٨ كيلوبايت بينما تسمى الذاكرة التى سعتها حوالى ٢٫٠٠٠٫٠٠٠ بايت ذاكرة لها ٢ ميجابايت . ومجموعات الاقراص النمطية يمكن ان تخزن مابين ٢٩ الى ١٢٦٠ ميجابايت طبقا لطرازها.

يسهم السريان الدائم للمعلومات من وحدات المدخلات او التخزين المساعد الى الذاكرة وتشغيل المعلومات اثناء وجودها فى الذاكرة والنقل النهائى للنتائج الى وحدات المخرجات او التخزين المساعد - فى خاصية دورة مدخلات - تشغيل - مخرجات in-put-process-output cycle لكل تشغيل بيانات الاعمال تقريبا .

## ١ - ٦ وحدة الحساب والمنطق ALU

وحدة الحساب والمنطق ( ALU ) هي جزء من وحدة التشغيل المركزية CPU ، التى تقوم بالتشغيل الفعلى للمعلومات . تستقبل وحدة الحساب والمنطق البيانات التى يجرى عليها التشغيل من الذاكرة . وتنفذ وحدة الحساب والمنطق نوعين اساسيين من تشغيل البيانات .

**الحساب** : عمليات الجمع والطرح والضرب والقسمة

**المنطق** : الوظيفة التى تقارن فيها وحدة الحساب والمنطق القيم المخزنة فى موقعى ذاكرة مختلفين لتحديد ما إذا كانت القيمتان متساويتين ، فاذا لم يكن الحال كذلك، فأيهما اكبر من الاخرى . ويمثل هذا القرار البسيط اساس كل انشطة الكمبيوتر المعقدة والتى تبدو ذكية .

## ١ - ٧ وحدة التحكم CU

تعمل اجهزة الكمبيوتر الحديثة بمفهوم البرنامج المخزن stored program الذى يسمح لنا بتغيير النشاط الذى يؤديه الكمبيوتر ( مثل الرواتب او المخزون ) ببساطة، وذلك بادخال تعليمات متتالية ( برنامج ) مختلفة فى الذاكرة . وتصميم وحدة التحكم لتنفيذ execute البرنامج المخزن على النحو التالى :

( ١ ) أول شىء تفعله وحدة التحكم هو احضار fetch احدى التعليمات المخزنة، والتى يكون عنوانها فى منطقة تخزين خاصة تسمى مسجل عنوان تعليمات Instruction Address Register (IAR).

( ٢ ) تقوم وحدة التحكم بعد ذلك بفك شفرة decode هذه التعليمة التى احضرت من الذاكرة ، اى أنها تحلل التعليمة لتحديد المطلوب عمله ( اى عملية operation مطلوب ادائها )، وماهى البيانات المطلوب استخدامها فى هذه العملية ( ما عناوين الذاكرة للعوامل التى تجرى عليها العملية operation ) .

( ٣ ) تستبدل وحدة التحكم عند ذلك العنوان فى مسجل عنوان التعليمات بعنوان التعليمه المخزنة التالية .

( ٤ ) اخيرا .. ترسل وحدة التحكم اشارات الى بقية النظام للتاكد ان العملية المحددة قد نفذت .

اى بالنسبة لعملية جمع ترتب وحدة التحكم للذاكرة بان ترسل نسخا من القيم المراد جمعها الى وحدة الحساب والمنطق . وتتسبب عند ذلك فى ان تنفذ وحدة الحساب والمنطق عملية الجمع فعلا وترسل حاصل الجمع الى موقع الذاكرة المطلوب . ولاحدى عمليات المدخلات او المخرجات تنشط وحدة المدخلات او المخرجات المحددة والذاكرة متسببة فى النقل المناسب للبيانات .

عندما تتم هذه المراحل الاربع لعمل وحدة التحكم ، تعاد الدورة ببساطة مرة اخرى من الخطوة الاولى . (يعاد اعداد مسجل عنوان التعليمات فى الخطوة الثالثة).

يستطيع المبرمج ان يغير من هذا التنفيذ المتتابع sequenttial executiaon المعتاد، وذلك بوضع تعليمات فرعية branch in-structions فى برنامجه . تغير التعليمات ؛ الفرعية من مسجل عنوان التعليمات بحيث انه يحتوى على عنوان غير عنوان التعليمه الطبيعية التالية ، ويتسبب ذلك فى بدء الكمبيوتر التنفيذ المتتالى الطبيعى عند العنوان الفرعى المحدد .

وتقيس السرعة التى يمكن لاجهزة الكمبيوتر الحديثة ان تنفذ بها التعليمات بعدد ملايين التعليمات المنفذة فى الثانية الواحدة millions instructions per second ( MIPS ) . وتعمل اجهزة الكمبيوتر المتاحة فى الاسواق حاليا بسرعات تتراوح من ٢ر٠ الى ١١ مليوناً من التعليمات فى الثانية الواحدة ومن المتوقع ان تتزايد هذه القيم باستمرار .

## ١ - ٨ لغات البرمجـــة

كما تخزن تعليمات البرنامج فى ذاكرة الكمبيوتر، وتنفذ فى وحدة التشغيل المركزية فإن لها شكل سلسلة من الأرقام الثنائية، ونقول انها معبر عنها بلغة الآلة machine language . الا ان الانسان يجد ان كتابة ( وقراءة ) البرامج باحدى اللغات مرتفعة المستوى high-level language مثل الكوبل مريحة جدا . ويؤدى الكمبيوتر بنفسه الترجمة اللازمة لبرنامج مكتوب بلغة مرتفعة المستوى ( برنامج المصدر source program ) الى مايكافئه من لغة الآلة ( برنامج الهدف object progcam ) وذلك تحت تحكم برنامج خاص يسمى بالمترجم Compiler . وعادة ماتكتب الشركة التى تنتج نظم المكونات المترجمات ، وتحفظ المترجمات فى التخزين المساعد بلغة الآلة .

تمثل عملية الترجمة ( من برنامج المصدر المكتوب بالكوبل ) فى شكل ١ - ٣ . اذا اكتشف المترجم اخطاء فى هيكل (اخطاء تكوينية Syntax errors) برنامج المصدر فانه يتسبب فى طباعة رسائل خطأ تشخيصية diagnostic error messages فى قائمة برنامج المصدر .

اذا اكتشفت اخطاء تكوينية، فيجب ان يصححها المبرمج ويعيد ترجمة برنامج المصدر المعدل . وإذا لم تكتشف أخطاء فى هذا البرنامج ( برنامج المصدر ) الجديد ، او برنامج المصدر الاصلى، يجب ان يصحح debugged ؛ اى يجب ان ينفذ بمجموعة من بيانات المدخلات تكون مخرجاتها الصحيحة معروفة مسبقا . وتصمم هذه العملية لاكتشاف اى أخطاء منطقية logic errors موجودة فى البرنامج أي تخطر الآلة بان تضيف ( بدلا من الضرب ) ساعات العمل ومعدل الاجر فى الساعة إلى حساب اجمالى الاجر . اثناء التصحيح.. يجب ان تعاد ترجمة برنامج المصدر فى كل مرة يحدث فيها تعديل له وذلك لحذف الاخطاء المنطقية .

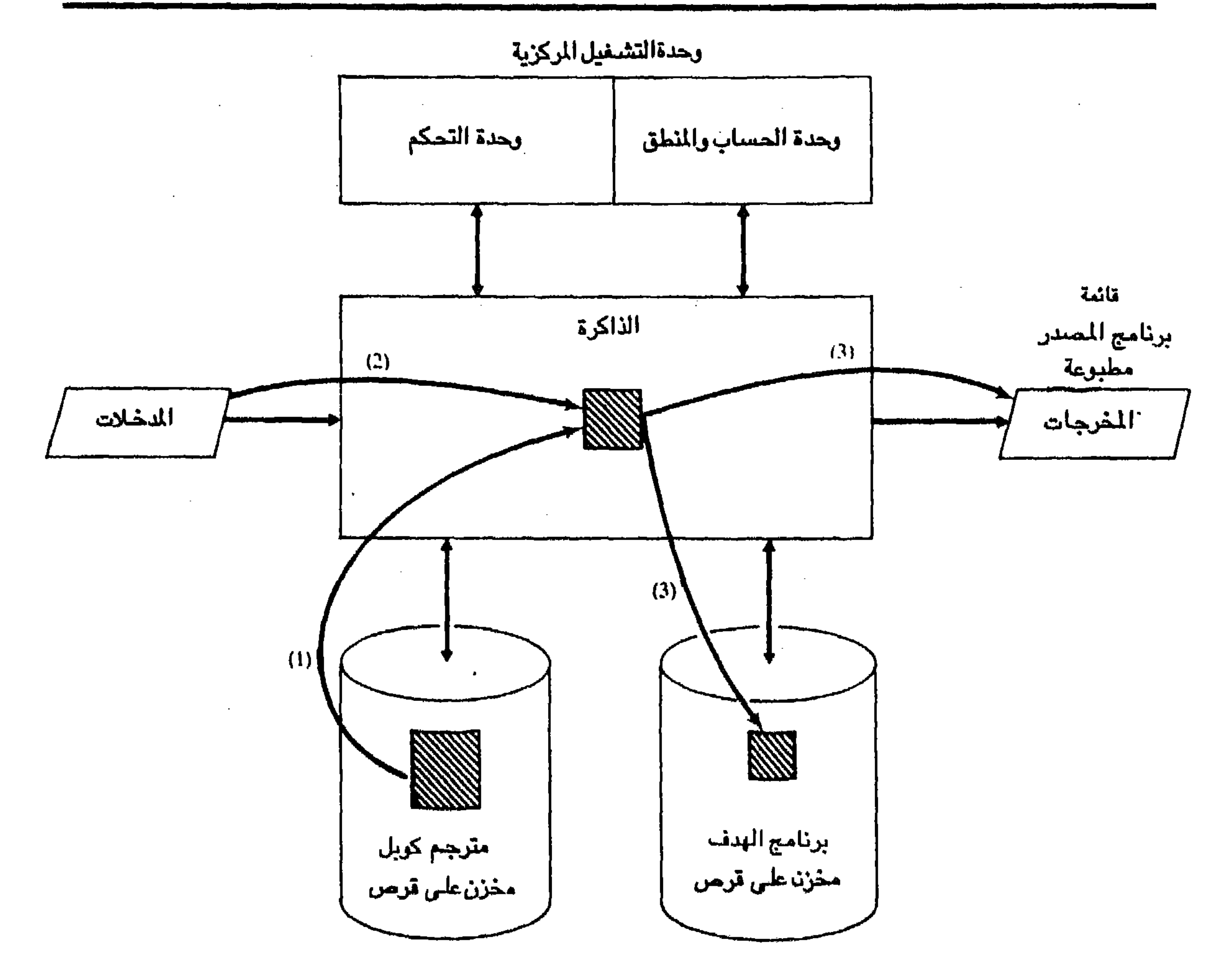
بعد اتمام تصحيح البرنامج.. تحفظ صيغة برنامج الهدف بصفة دائمة فى التخزين المساعد بحيث يمكن قراءته داخل الذاكرة وتنفيذه كلما كانت هناك حاجة لذلك .

## ١ - ٩ نظـــم التشـــغيل

بدلا من جعل المشغل الآدمى opeator مسؤلا عن تحميل loading برنامج التنفيذ داخل الذاكرة لتنفيذه.. قام منتجو الكمبيوتر والمتخصصون الآخرون بكتابة برامج تسمى نظم تشغيل operating systems ، التى تسمح للكمبيوتر بعمل هذا النشاط بنفسه وبسرعات الآلة . وتخزن برامج نظام التشغيل هذه بصفة دائمة بلغة الآلة على القرص .

يحتفظ مشغلو الكمبيوتر الآدميون بتحكمهم فى نظام الكمبيوتر عن طريق ادخال معلومات لنظام لتشغيل تذكر له اى برامج اخرى تحمل وتنفذ . وعادة .. مايحدث ذلك بطريقتين . بعض نظم التشغيل مصممة بحيث تقبل عبارات خاصة تسمى لغة تحكم العمل ( job control language ( JCL من اى وحدة مدخلات . تذكر عبارات لغة تحكم العمل لنظام التشغيل محتويات برامج التطبيقات application programs ( مثل : المخزون والرواتب وحسابات المدينين وحسابات الدائنين ) التى تحمل داخل الذاكرة. ويصمم بعضها الاخر ليسمح بعلاقة تحاور conversational ( او تداخل interactive ) بين نظام التشغيل والانسان الموجود امام الشاشة .

يستطيع الانسان كتابة اوامر commands ينفذها نظام التشغيل على الفور . وتذكر الاوامر للنظام البرامج التى يريد المستفيد تحميلها وتنفيذها . ولممارسة البرمجة بالكوبل - بالفعل - على جهاز كمبيوتر .. يجب ان تتعلم اولا لغة تحكم العمل او لغة الاوامر الخاصة بنظام تشغيل الجهاز .




( ١ ) يتم إحضار المترجم من التخزين المساعد الى داخل الذاكرة

( ٢ ) ينفذ الكمبيوتر المترجم متسببا فى قراءة وترجمة برنامج المصدر المكتوب بالكوبل

( ٣ ) تتسبب تعليمات المترجم فى وضع برنامج الهدف على القرص ( لاستخدامه فيما بعد ). وطباعة قائمة ببرنامج المصدر.

شكل (١ - ٣)

## ١ - ١٠ الملفــــات

تحفظ كل من البرامج والبيانات فى التخزين المساعد على هيئة ملفات. ويحتوى الملف file على معلومات عن برنامج خاص او تطبيق خاص ويعرف بـ «اسم الملف» file name . فمثلاً.. يمكن للملف المسمى JAN. PAYROLL ان يحتوى على معلومات عن حسابات المدينين . وتعتمد القواعد الدقيقة لعمل اسماء الملفات على نظام التشغيل الخاص بجهاز الكمبيوتر .

تنقسم الملفات الى سجلات ، يحتوى السجل منها record على معلومات عن شخص معين او شيء معين . إذ، يمكن ان يوجد بالملف المسمى EMPLIYEE. DATA سجل معلومات واحد عن شخص اسمه Smith وسجل معلومات آخر عن شخص اسمه Jones . والسجلات الموجودة على شريط او على قرص تفصل عن بعضها البعض واقعيا بواسطة مناطق فارغة تسمى بفراغات مابين السجلات interrecord gaps .

وتنقل مشغلات الشرائط والاقراص الخاصة بالسجلات الى ذاكرة الكمبيوتر سجلاً واحدا فى نفس الوقت one record at a time . وعلى هذا فعندما يقوم برنامج الكوبل بتشغيل ملف فيجب ان يعمل مع سجل واحد فى نفس الوقت . ويجب ان يفكر المبرمج فى مصطلحات دورة المدخلات والتشغيل والمخرجات : اى ادخال سجل من الملف وتشغيل السجل واخراج النتائج وتكرار ذلك حتى يتم تشغيل جميع السجلات .

وينقسم السجل الى حقول ، يعد الحقل field الواحد منها قطعة محددة من المعلومات ، مثل الاسم او رقم الحساب، او اجمالى الراتب او غيرها . ويتكون كل حقل بدوره من مجموعة من البايت byte الفردية ( او الرموز الفردية ) التى تعطى معناه . وتوضح هرمية الملف والسجل والحقل والرمز فى شكل (١ - ٤) .



شكل (١ - ٤)

تحتفظ معظم نظم التشغيل بعنوان ملف file label لكل ملف على شريط او على قرص . وعنوان الملف هو سجل خاص يحتوى على حقول مثل : اسم الملف الذى يحدد عند انتاج الملف ، وطول الملف وموقع الملف ( على القرص ) وحجم الملف والمعلومات الخاصة بكلمة المرور التى تؤمن الملف من الاتصال غير المسئول ... الخ . وتسبق عناوين ملف الشريط محتويات الملف مباشرة كما تتبعها مباشرة ، وعادة ما تجمع عناوين ملف القرص مع بعضها فيما يسمى جدول محتويات الحجم volume table of contents (VTOC) او دليل directory . ويستطيع نظام التشغيل تحديد موقع اى ملف موجود على مجموعة الاقراص بالبحث عن عنوانه فى الدليل . ويساعد استخدام العناوين فى التاكد من ان البرامج تقوم بتشغيل الملفات الصحيحة.

## ١ - ١١ شفرات البيانات

الوحدة الاساسية للذاكرة وللتخزين المساعد هى ، كما نعرف ، الرقم الثنائى او البت . ويجب أن تكتب أى معلومات مخزنة فى هذه الاجزاء من النظام بشفرة coded كسلسلة من الارقام الثنائية. وبهدف عمل الشفرة فمن المقنع تقسيم البيانات كما يلى :

( ١ ) **بيانات حرفية عددية** alphanumeric أو (alphameric) يمكن ان تتكون من اى رموز من اى لوحة مفاتيح للمدخلات او اى شاشة مرئية للمخرجات او اى طابع . وتشمل هذه الفئة كل الحروف letter الهجائية والارقام digits من ٠ الى ٩ وكل الرموز الخاصة special characters مثل : (@ , # , $ , % , & , *) و (-و + الفراغ .. الخ ) الحقل "John W. Doe" هو حقل حرفي عددي ( لاحظ الرمز الخاص ، النقطة ) كما ان الحقل «٩٧٦-٢٣٨» هو حقل حرفي عددي كذلك (لاحظ الرمز الخاص الشرطة).

( ٢ ) **بيانات حرفية** alphabetic تتكون من الحروف الهجائية والفراغ . الحقل"John W Doe" هو حقل حرفي ، وكذلك الحقل "N SPINT ST" هو حقل حرفي ، اما الحقلان "John W. Doe' و "43 N SPRINT ST" فليسا حرفيين .

( ٣ ) **بيانات عددية** numeric تتكون من الارقام من ٠ الى ٩ مع وجود اشارة سالب او موجب اختيارية . فى بعض الحالات يمكن ان تحتوى البيانات العددية على علامة عشرية ، الا انه عادة ما يفترض assumed وجود العلامة العشرية ومن أمثلة البيانات العددية مايلي : "1278" ، "53 -" ، "1.28" ، "1‸28" ؛ حيث يمثل الرمز ‸ علامة عشرية مفترضة .

تستخدم الشفرات codes التالية فى تمثيل بيانات حرفية عددية، وحرفية وعددية كاعداد ثنائية مناسبة لتخزينها فى نظام الكمبيوتر .

( ١ ) EBCDIC. DISPLAY (EBCDIC & ASCII) هى شفرة التبادل العشرية المتسعة للشفرة الثنائية Extended و Binary- Coded Decimal Interchange Code , ASCII هى الشفرة الامريكية النمطية لتبادل المعلومات American Sta dard Code For Information Interchange ، وهما اكثر شفرتين مستخدمتين فى تمثيل البيانات الحرفية عددية. وتمثل كل من الشفرتين كل رمز حرفي عددى ( بما فى ذلك الفراغ ) برقم ثنائى له ٨ بايت ( حيث يتطلب كل رمز بايت واحد من الذاكرة او التخزين المساعد ) . والشفرة الكاملة مسرودة فى ملحق ب .

( ٢ ) COMP-3 (packed-decimal) . العشرى المضغوط Packed-decimal هى شفرة تستخدم على نحو صارم مع البيانات العددية فى الذاكرة او التخزين المساعد ولاتستخدم بواسطة اى وحدة مدخلات او مخرجات ، كما انها ليست متاحة على كل اجهزة الكمبيوتر. ويقال فى الكوبل – عن اعداد العشرى المضغوط – بانها صورة COMP-3. ولشفرة العشرى المضغوط المميزات التالية عن شفرة EBCDIC ، وشفرة ASCII :

( أ ) يمكن لوحدة الحساب والمنطق اجراء الحسابات على اعداد العشرى المضغوط مباشرة (اذا دخلت الاعداد الحسابات وهى فى شفرة EBCDIC او شفرة ASCII إذ يجب ان يحو لها برنامج التنفيذ الى العشرى المضغوط مؤقتا لاجراء الحسابات ثم يعيد تحويل النتيجة الى شفرة EBCDIC او شفرة ASCII) .

(ب ) تشغل اعداد العشرى المضغوط عددا اقل من البايت عن نفس الاعداد الممثلة بشفرة EBCDIC او شفرة ASCII.

( ٣ ) COMP (binary-twos-complement) . مكمل الازواج الثنائية ، وله نفس التطبيق والمميزات التى للعشرى المضغوط ، ويقال فى الكوبل عن الاعداد الممثلة بمكمل الازواج الثنائية بانها فى صورة COMP.

وكل كمبيوتر تقريبا يدعم بعض صيغ مكمل الازواج الثنائية ، كما ان بعضها يدعم العشرى المضغوط كذلك. انظر الفصل الخامس لمناقشة وقت استخدام اي من الشفرات السابقة.

# الفصل الثانى

# عـــرض ســـريع للكـــوبل

# Cobol Overview

إحدى سمات الكوبل المهمة هى التوثيق الذاتى self- documenting : يستطيع المبرمج أن يفهم معظم أجزاء برنامج الكوبل بقراءة شفرة البرنامج ببساطة . افحص برنامج الكوبل الكامل الموجود فى مثال (٢ - ١) الذى تتكرر الإشارة إليه فى هذا الفصل .

**مثـــال ٢ - ١ :**

```
 IDENTIFICATION DIVISION.

 PROGRAM-ID. QUARTER.
 AUTHOR. LARRY NEWCOMER.
 INSTALLATION.  PENN STATE UNIVERSITY--YORK CAMPUS.
 DATE-WRITTEN.  MAY 1983.
 DATE-COMPILED. MAY  9,1983.
 SECURITY.  THERE ARE NO SECURITY CONSIDERATIONS FOR QUARTER.
*
*OVERVIEW OF PROGRAM QUARTER--
*
*    QUARTER READS A FILE CONTAINING SALESPERSON MONTHLY
*    SALES FOR THE 3 MONTHS IN A QUARTER ALONG WITH THE
*    SALESPERSON'S NAME AND QUARTERLY QUOTA SALES AMOUNT.
*    IT PRINTS A REPORT SHOWING SALESPERSON:
*
*         NAME     QUARTERLY SALES       QUOTA
*
*         JONES       $42,000.98        $40,000.00
*         SMITH       $59,000.67        $60,000.00
*         YOST        $47,893.00        $45,000.00
*

 ENVIRONMENT DIVISION.

 CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-370.
 OBJECT-COMPUTER. IBM-370.

 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT SALES-FILE        ASSIGN TO SALES.
     SELECT QUARTERLY-REPORT ASSIGN TO SALESRPT.
```

```
       DATA DIVISION.

       FILE SECTION.

       FD SALES-FILE
           LABEL RECORDS ARE STANDARD
           RECORD CONTAINS 80 CHARACTERS
           .
       01  SALES-RECORD.
           05  SALES-RECORD-NAME            PIC X(15).
           05  SALES-RECORD-MONTH-1-SALES   PIC S9(4)V99.
           05  SALES-RECORD-MONTH-2-SALES   PIC S9(4)V99.
           05  SALES-RECORD-MONTH-3-SALES   PIC S9(4)V99.
           05  SALES-RECORD-QUOTA.          PIC S9(5)V99.
           05  FILLER                       PIC X(40).


       FD  QUARTERLY-REPORT
           LABEL RECORDS ARE OMITTED
           RECORD CONTAINS 132 CHARACTERS
           .
       01  QUARTERLY-REPORT-LINE            PIC X(132).

       WORKING-STORAGE SECTION.

       01  SWITCHES-AND-TOTALS.
           05  SALES-FILE-END               PIC X.
           05  QUARTERLY-TOTAL              PIC S9(5)V99     COMP-3.
       01  WORKING-REPORT-LINE.
           05  WORKING-NAME                 PIC X(15).
           05  FILLER                       PIC X(5)    VALUE SPACES.
           05  WORKING-TOTAL                PIC $$$,$$$.99.
           05  FILLER                       PIC X(5)    VALUE SPACES.
           05  WORKING-QUOTA                PIC $$$,$$$.99.
           05  FILLER                       PIC X(87)   VALUE SPACES.

       PROCEDURE DIVISION.

       010-EXECUTIVE-PARA.
           PERFORM 020-INITIALIZE-AND-SET-UP
           PERFORM 040-PRINT-REPORT-LINES
               UNTIL SALES-FILE-END = "T"
           PERFORM 050-TERMINATION-AND-WIND-UP
           STOP RUN
           .

       020-INITIALIZE-AND-SET-UP.
           OPEN  INPUT  SALES-FILE
                 OUTPUT QUARTERLY-REPORT
           MOVE "F" TO SALES-FILE-END
           PERFORM 030-READ-SALES-FILE
           .

       030-READ-SALES-FILE.
           READ SALES-FILE
               AT END
                   MOVE "T" TO SALES-FILE-END
```

```
00093          040-PRINT-REPORT-LINES.
00094              MOVE SALES-RECORD-NAME TO WORKING-NAME
00095              COMPUTE QUARTERLY-TOTAL =   SALES-RECORD-MONTH-1-SALES
00096                                        + SALES-RECORD-MONTH-2-SALES
00097                                        + SALES-RECORD-MONTH-3-SALES
00098              MOVE QUARTERLY-TOTAL TO WORKING-TOTAL
00099              MOVE SALES-RECORD-QUOTA TO WORKING-QUOTA
00100              WRITE QUARTERLY-REPORT-LINE
00101                  FROM WORKING-REPORT-LINE
00102              PERFORM 030-READ-SALES-FILE
00103
00104
00105          050-TERMINATION-AND-WIND-UP.
00106              CLOSE  SALES-FILE
00107                     QUARTERLY-REPORT
00108
```

## ٢ - ١ أجـــزاء الكوبـــل

يحتوى الكوبل دائما على أربعة أجزاء رئيسية DIVISIONS ، وتكون بنفس الترتيب التالى : التعريف IDENTIF-ICATION ، ثم الأوساط ENVIRONMENT ، ثم البيانات DATA ، ثم الإجراءات PROCEDURE . وفى بداية كل جزء .. توجد عبارة كوبل خاصة تسمى عنوان الجزء division header .

**مثـــال ٢ - ٢ :**

عناوين الأجزاء فى مثال (٢ - ١) هى :

السطر رقم ١ : IDENTIFICATION DIVIDION جزء التعريف .

السطر رقم ٢٤ : ENVIRONMENT DIVISION جزء الاوساط .

السطر رقم ٣٥ : DATA DIVISION جزء البيانات.

السطر رقم ٧٠ : PROCEDURE DIVISION جزء الإجراءات .

يقدم جزء التعريف اسم البرنامج ، ومن الذى كتبه، ومتى كتب وأين كتب، ومتى تمت ترجمته، وماهى احتياطات الأمن (إذا كانت هناك مثل هذه الاحتياطات) التى يجب اتخاذها لتقييد الاتصال بالبرنامج، وبالملفات التى يقوم بتشغيلها . ويجب أن ينتهى هذا الجزء بمجموعة جمل إنجليزية تعطى عرضا سريعا لما يفعله البرنامج (الأسطر من ٩ - ٢٢ موضحة فى مثال (٢ - ١) ).

يحتوى جزء الأوساط على معلومات عن الكمبيوتر الذى يترجم عليه برنامج الكوبل، والذى ينفذ عليه البرنامج المكتوب بلغة الآلة، والناتج من عملية الترجمة . كما يعطى كذلك اسم كوبل لكل ملف يراد تشغيله، ويحدد وحدة مدخلات أو وحدة مخرجات لكل ملف .

تعرف هذه المعلومات وسط نظم المكونات hardware environment الذى يتم تنفيذ البرنامج فيه .

**مثـــال ٢ - ٣ :**

فى مثال (٢ - ١) :

• السطر رقم ٢٧ : SOURCE-COMPUTER. IBM-370 يحدد الكمبيوتر المستخدم فى برنامج المصدر .

• السطر رقم ٢٨ : OBJECT-COMPUTER. IBM-370 يحدد الكمبيوتر المستخدم فى تنفيذ الهدف .

• السطران ٣٢ و ٣٣ : SELECT SALES-FILE ASSIGN TO SALES

.SELECT QUARTERLY- REPORT ASSIGN TO SALESRPT

يحددان وحدات المدخلات والمخرجات للملفات .

يعطى جزء البيانات وصفا موجزا لكل ملف يجرى عليه تشغيل، ويعطى كذلك تخطيطا تفصيليا للسجلات الموجودة فى الملف يوصف كل حقل فى سجل بالنسبة لطوله ونوع بياناته. وكذلك يصف جزء البيانات حقول البيانات التى يستخدمها البرنامج ، ولاتظهر فى سجلات الملفات . مثل هذه الحقول.. تخزن فى مناطق ذاكرة ، تسمى مخزن العمل WORKING-STORAGE، ويجب أن توصف كل حقول البيانات (فى سجلات الملف، أو فى مخزن العمل) فى جزء البيانات .

**مثال ٢ ـ ٤ :**

تصف الأسطر من ٣٩ – ٤٩ فى مثال (٢ – ١) ملف المدخلات للمبيعات والسجلات التى يحتويها .

```
FD  SALES-FILE
    LABEL RECORDS ARE STANDARD
    RECORD CONTAINS 80 CHARACTERS
    .
01  SALES-RECORD.
    05  SALES-RECORD-NAME              PIC X(15).
    05  SALES-RECORD-MONTH-1-SALES     PIC S9(4)V99.
    05  SALES-RECORD-MONTH-2-SALES     PIC S9(4)V99.
    05  SALES-RECORD-MONTH-3-SALES     PIC S9(4)V99.
    05  SALES-RECORD-QUOTA             PIC S9(5)V99.
    05  FILLER                         PIC X(40).
```

تصف الأسطر من ٥٧ – ٦٨ عناصر بيانات مخزن العمل ، التى لاتكون جزءاً من أى سجلات للفات .

```
WORKING-STORAGE SECTION.

01  SWITCHES-AND-TOTALS.
    05  SALES-FILE-END                 PIC X.
    05  QUARTERLY-TOTAL                PIC S9(5)V99    COMP-3.
01  WORKING-REPORT-LINE.
    05  WORKING-NAME                   PIC X(15).
    05  FILLER                         PIC X(5)  VALUE SPACES.
    05  WORKING-TOTAL                  PIC $$$,$$$.99.
    05  FILLER                         PIC X(5)  VALUE SPACES.
    05  WORKING-QUOTA                  PIC $$$,$$$.99.
    05  FILLER                         PIC X(87) VALUE SPACES.
```

وفى جزء الإجراءات .. يتم إخطار الكمبيوتر فعلا بالتشغيل المراد أدائه . وتشبه التوجيهات الجمل الإنجليزية بدرجة كبيرة

**مثــال ٢ - ٥ :**

عبارات جزء الإجراءات التالية مأخوذة من مثال ٢ - ١ .

• PERFORM 020- INITIALIZE- AND- SET- UP

(السطر ٧٣) يتسبب فى تنفيذ كل العبارات الموجودة فى المقطع المسمى 020- INITIALIZE- AND- SET- UP ، وهى الأسطر من ٨٠ - ٨٥، وذلك مرة واحدة قبل أن يستمر الكمبيوتر فى تنفيذ العبارة التالية لعبارة :

PERFORM 020- INITIALIZE- AND- SET- UP

(أى السطر رقم ٧٤)

• PERFORM 040- PRINT- REPORT- LINES- UP UNTIL SALES- FILE- END= " T "

(السطران ٧٤ - ٧٥) يتسببان فى أن ينفذ الكمبيوتر العبارات الموجودة فى المقطع المسمى 040- PRINT- REPORT- LINES . الأسطر من (٩٣ - ١٠٣) وذلك مرات ومرات على التوالى، حتى يحتوى عنصر البيانات المسمى SALES- FILE- END على القيمة " T ". عندما يحدث ذلك .. ينتقل الكمبيوتر إلى العبارة التالية (أى السطر ٧٦) .

• OPEN INPUT SALES- FILE OUTPUT QUARTERLY- REPORT .

(السطران ٨١ - ٨٢) تعد عبارة OPEN الملف للتشغيل، وتحدد ما إذا كان الملف سيستخدم كمدخلات أو مخرجات، ويجب أن يفتح الملف قبل أن يستخدم فى البرنامج.

• MOVE SALES- RECORD- NAME TO WORKING- NAME .

(السطر٩٤) تنسخ عبارة MOVE محتويات أحد مواقع التخزين (فى هذه الحالة الموقع المحدد لعنصر البيانات المسمى SALES- RECORD- NAME فى موقع تخزين آخر (فى هذه الحالة الموقع المحدد لعنصر البيانات المسمى WORKING- NAME).

```
COMPUTE QUARTERLY-TOTAL = SALES-RECORD-MONTH-1-SALES
                        + SALES-RECORD-MONTH-2-SALES
                        + SALES-RECORD-MONTH-3-SALES
```

(الأسطر ٩٥ - ٩٧) تستخدم عبارة COMPUTE فى إجراء حسابات عددية، تخزن نتيجة الحسابات في عنصر بيانات (أى فى موقع الذاكرة المحدد لعنصر البيانات)، الذى يكتب اسمه فى الناحية اليسرى من علامة التساوى.

## ٢ - ٣ الأقســام والمقاطــع

يتكون كل جزء من أجزاء برنامج الكوبل من أقسام SECTIONS، تبدأ فى أسطر خاصة، تسمى عناوين الأقسام section headers، ويتكون كل قسم بدوره من مقاطع paragraphs.

**مثــال ٢ - ٦ :**

عناوين الأقسام فى مثال ٢ - ١ هى :

السطر رقم ٢٦ : CONFIGURATION SECTION ، قسم التشكيل.

السطر رقم ٣٠ : INOUT- OUTPUT SECTION ، قسم المدخلات والمخرجات .

السطر رقم ٣٧ : FILE SECTION ، قسم الملفات .

السطر رقم ٥٧ : WORKING- STORAGE SECTION ، قسم مخزن العمل .

تحتوى عناوين الأقسام على اسم القسم، تتبعه كلمة SECTION ثم نقطة. وتعد أسماء الأقسام في جزء الأوساط وجزء البيانات، جزءاً ثابتاً من لغة الكوبل، لايمكن تغييرها (مثل هذه الكلمات تسمى كلمات محجوزة reserved words) . والأقسام اختيارية فى جزء الإجراءات، وعندما تستخدم.. يجب أن تكتب اسماؤها بنفس قواعد كتابة كلمات الكوبل (مثل الأسماء التى يعرفها المستفيد programmer- defined names) . ولاتوجد أى أقسام فى جزء التعريف .

يمكن أن تكون المقاطع أجزاء.. فى قسم، ويمكن أن تكون قائمة بذاتها داخل الجزء . وفى كل الحالات.. تعرف المقاطع بعنوان paragraph header، الذى يحتوى على اسم المقطع فقط تليه نقطة. أسماء المقاطع فى جزء التعريف، وجزء الأوساط، وجزء البيانات هو جزء ثابت من الكوبل، ولايمكن تغييرها . والمقاطع فى جزء الإجراءات اختيارية، ويضع المبرمج أسماءها .

**مثـال ٢ - ٧ :**

يتكون جزء التعريف من مقاطع فقط (لاتوجد به أقسام). أسماء المقاطع المستخدمة فى جزء التعريف فى المثال ٢ - ١ هى:

PROGRAM-ID AUTHOR INSTALLATION DATE-WRITTEN DATE-COMPILED
SECURITY

أسماء المقاطع جزء ثابت من لغة الكوبل.

**مثـال ٢ - ٨ :**

يمكن أن يتكون جزء الإجراءات من برنامج الكوبل، أو لايتكون، من أقسام. ويستخدم مثال (٢ - ١) مقاطع فقط فى جزء الإجراءات. والأسماء التى عرف المستفيد لها هى :

010-EXECUTIVE-PARA 020-INITIALIZE-AND-SET-UP 030-READ-SALES-FILE
040-PRINT-REPORT-LINES 050-TERMINATION-AND-WIND-UP

وكذلك بالرغم من أنه ليس مطلوبا في لغة الكوبل أن تبدأ أسماء المقاطع بأرقام إلا أن أسماء كل هذه المقاطع بدأت بأرقام، وهذه طريقة تجعل الوصول إلى مقطع معين فى البرنامج أسهل بكثير (بافتراض أن الأرقام تظهر بالترتيب)

## استخدام النقطـــة

تتكون المقاطع فى جزء الإجراءات من جمل، مثل اللغة الانجليزية، ويعرف المستفيد الجمل باستخدام نقاط. وبالرغم من أن الكوبل يسمح بوجود نقطة فى نهاية كل جملة، إلا أنها فكرة جيدة أن تستخدم النقاط فى جزء الإجراءات عند الحاجة الضرورية فقط. لاحظ في جزء الإجراءات من مثال ١ – ٢ أن النقاط أسطر خاصة. ويوصى بهذا الاستخدام للنقطة (النقطة المرتبة structured period)؛ نظراً لأنه يجعل النقطة مرئية بوضوح، مع السماح بإدخال عبارات جديدة أمام النقطة، ويساعد ذلك فى تقليل الأخطاء المنطقية.

**مثـال ٢ – ٩ :**

يجب استخدام النقطة فى الكوبل لإنهاء كل مقطع، كما تكون هناك حاجة لها فى بعض الأحيان لفصل عبارة عن عبارة أخرى داخل نفس المقطع.

يستخدم جزء الإجراءات فى مثال (٢ – ١) (١) نقطة بعد عنوان الجزء (السطر ٧٠)، و(٢) نقطة بعد عنوان كل مقطع (الاسطر رقم ٧٢ و٨٠ و٨٧ و٩٣ و١٠٥) و(٣) نقطة مرتبة فى نهاية كل مقطع (الأسطر : ٧٨ و٨٥ و٩٣ و١٠٣ و١٠٨).

**مثـال ٢ – ١٠ :**

يحتوى السطران : (٤٢ ، ٥٤) فى مثال (٢ – ١) علي نقاط مرتبة. وهى أجزاء من جزء البيانات، قد تتطلب حذف أو اضافة لأسطر كوبل، مع إجراء التغييرات على البرنامج ، وتسهل النقطة إجراء مثل هذا التغييرات.، إلا أن القاعدة المعتادة هى وضع النقاط، خارج جزء الإجراءات، على نفس السطر الذى تنهيه .

## ٢ – ٣ قواعد كتابة برامج كوبل

عادة مايكتب مبرمجو الكوبل برامجهم على صيغة كتابة شفرة coding forms خاصة بالكوبل (شكل ٢ – ١ ) . وتستخدم الأعمدة من ١ – ٦ من كل سطر فى كتابة أرقام متتالية تيسر اعادة ترتيب البرنامج إذا حدث خلط لأسطره بطريق الخطأ (خلط للبطاقات المثقبة) . تستخدم الأعمدة من العمود رقم ٧٣ الى العمود ٨٠ من كل سطر فى تعريف البرنامج. فيكتب اختصار لاسم البرنامج فى هذه الأعمدة. الأرقام المتتالية والتعريف مهمة جدا، عندما يثقب البرنامج في بطاقات، لان مجموعة البطاقات يمكن أن تقع. عادة ماتكتب البرامج الآن عن طريق نهاية طرفية، ملحق بها شاشة مرئية ويخزن على قرص، وهذا يقلل الأهتمام بالتتابع والتعريف.

تكتب عبارات الكوبل نفسها فى الأعمدة من العمود الثامن وحتى العمود الثانى والسبعين. تمثل الأعمدة من (٨ – ١١) المنطقة A، وتمثل الأعمدة من العمود (١٢ – ٧٢) المنطقة B . والعناصر التالية يجب أن تبدأ كتابتها فى المنطقة A.

- عناوين الأجزاء (أسماء الأجزاء) .
- عناوين الاقسام (أسماء الأقسام) .
- عناوين المقاطع (أسماأء المقاطع) .

وداخل جزء البيانات.

- عناوين محتويات وصف الملف (بدءا بالأحرف FD ، كما فى الأسطر ٣٩ – ٥١ من مثال (٢ – ١).
- عناوين محتويات وصف السجل (عناصر البيانات التى لها رقم مستوي 01) ، كما فى الأسطر رقم ٤٣، ٥٥، ٥٩، ٦٢ فى

مثال ٢ – ١١. يجب أن تبدأ كتابة جميع العناصر الأخرى فى المنطقة B . وبالرغم من أن الكوبل يسمح بمحتويات المنطقة B ، لأن تبدأ من العمود رقم ١٢ ... إلا أنه من المهم عمل ترحيل فى البرمجة المرتبة داخل المنطقة B لتوضيح العلاقات بين الأسطر.

| SYSTEM | | PUNCHING INSTRUCTIONS | | | PAGE OF |
|---|---|---|---|---|---|
| PROGRAM | | GRAPHIC | | | CARD FORM # |
| PROGRAMMER | DATE | PUNCH | | | * |

| SEQUENCE (PAGE) (SERIAL) | CONT. | A | B | COBOL STATEMENT | IDENTIFICATION |
|---|---|---|---|---|---|
| 1 3 4 6 | 7 | 8 | 12 | 16 20 24 28 32 36 40 44 48 52 56 60 64 68 72 | 76 80 |

شكل (٢ - ١)

**مثـال ٢ - ١١ :**

فى عبارات جزء الإجراءات التالية من مثال ( ٢ - ١) تكون الأسطر التى تستمر فيها العبارة - التى بدأت في السطر السابق - مرحلة تحت السطر الأصلى .

الأسطر من ٨٨ - ٩٠ :

```
READ SALES-FILE
    AT END
        MOVE "T" TO SALES-FILE-END
```

السطران ١٠٠ - ١٠١ :

```
WRITE QUARTERLY-REPORT-LINE
    FROM WORKING-REPORT-LINE
```

## استخدامات خاصة للعمود السابع

يفترض الكوبل أن العبارة تستمر ، سطرا بعد سطر، حتى تظهر نقطة أو عبارة أخرى. وهناك حاجة إلى قواعد خاصة للأستمرارية عندما ينتهى السطر فى منتصف الكلمة، أو فى منتصف سلسلة موضوعة بين علامتى تنصيص quoted string فقط (القسم التاسع عشر من الفصل الخامس) :

(١) إذا انتهى السطر السابق فى منتصف الكلمة ... اكتب شرطة « - » فى العمود السابع من السطر التالى، واستمر فى كتابة بقية الكلمة في المنطقة B من السطر التالى .

(٢) إذا انتهى السطر السابق فى منتصف سلسلة موضوعة بين علامتى تنصيص ... اكتب « - » فى العمود السابع من السطر التالى، واكتب علامة تنصيص فى أى مكان فى المنطقة B من السطر التالى، واستمر بعدها فى كتابة بقية السلسلة.

**مثـال ٢ - ١٢ :**

```
     MOVE "THANK YOU FOR YOUR
-    |       "PROMPT PAYMENT" TO
|    |       CUSTOMER-MESSAGE
|    |       ↑
col. col.  col.
7    12     16
```

بينما يعرض استمرار مناسب، فما سبق ذكره ينتج عنه فراغات عديدة غير مطلوبة ؛ لان أول سطر من السلسلة الموضوعة بين علامتى تنصيص يفترض أستمراره حتى العمود ٧٢ (ليس أكثر من ذلك) .

**مثـال ٢ - ١٣ :**

ما الخطأ فى الاستمرار التالى :

```
     MOVE "OVER
-    |       "PAYMENT" TO NEGATIVE-BALANCE-MESSAGE
↑    |       ↑
col. col.  col.
7    12     16
```

(أ) كان يمكن تجنب الاستمرار فى المكان الأول

```
MOVE "OVERPAYMENT"
    TO NEGATIVE-BALANCE-MESSAGE
```

(ب) نظراً لأن السطر الأول يمتد على طول الطريق حتى العمود ٧٢... فيجب أن تظهر الرسالة فعلاً على النحو التالى:

```
"OVER                                   PAYMENT"
```

الشىء غير المطلوب

عندما تكتب نجمة (*) فى العمود السابع.. تعامل بقية السطر كتعليق comment كوبل. وهذا يعنى أنه بالرغم من أن المترجم يطبع السطر فى قائمة برنامج المصدر، إلا أنه لايترجمه إلى برنامج الهدف. وتستخدم التعليقات لتوضيح مايفعله البرنامج وكيفية أدائه لذلك، وهى موجهة إلى مبرمجى الكوبل الآخرين، الذين يحتاجون إلى إجراء تغييرات على البرنامج. تذكر: بما أن الكوبل لغة توثيق ذاتى.. فإن التعليقات الزائدة تستخدم فقط عندما تكون هناك حاجة حقيقية لتوضيحات أكثر.

**مثـــال ٢ - ١٤**

هناك تعليقات موضحة فى الأسطر من ٩ - ٢٢ من مثال (٢ - ١) .

يستخدم العمود السابع كذلك فى إدخال اسطر تصحيح debugging lines فى برنامج الكوبل، وهى ببساطة عبارات كوبل مع كتابة D فى العمود السابع . كما أسمها.. فهى تستخدم فى المساعدة في تحديد وتصحيح الأخطاء فى البرنامج (انظر القسم الثانى من الفصل الرابع) .

## الأســطر الفارغــة

يمكن أن تظهر اسطر فارغة فى برنامج الكوبل، وتستخدم لتسهيل قراءة برنامج المصدر (تجعل الاسطر رقم ٢ ، ٢٣، ٢٩... و١٠٤ مثال ٢ - ١ أسهل فى قراءته).

يفضل بعض المبرمجين أسطر تعليقات فارغة blank comment lines (أسطر موضوعة نجمة (*) فقط فى عمودها السابع) لأن المترجم يستطيع أن يشغل سطر التعليق أسرع من تشغيله السطر الفارغ كله، الا أن الميزة صغيرة جداً، وسوف نستخدم أسطر فارغة كلية.

## ٢ - ٤ الاسماء التى يعرفها المبرمج والكلمات المحجوزة

تتشكل الكلمات الفردية التى يتكون منها برنامج الكوبل طبقا للقواعد التالية :

( ١ ) لايزيد طولها عن ٣٠ رمزا.

( ٢ ) الرموز المسموح بها هى الحروف الهجائية من A الى Z ، والارقام من ٠ الى ٩ والشرطة .

( ٣ ) يجب الا تبدأ الكلمة او تنتهى بشرطة .

( ٤ ) يجب الا توجد فراغات داخل الكلمة .

تعتبر بعض الكلمات فى برنامج الكوبل خاصة بلغة الكوبل، وتخدم غرضاً واحداً فقط سبق تعريفه . هذه الكلمات المحجوزة reserved words تشمل : IDENTIFICATION و DIVISION و DATE-WRITTEN و INPUT-OUTPUT و SECTION و WORKING-STORAGE و MOVE و OPEN وCOMPUTE، ويوجد بملحق A قائمة كاملة بالكلمات المحجوزة . يجب ألا يكرر هجاء أى كلمة من الاسماء التى يعرفها المبرمج - والتى تعد طبقا للقواعد سالفة الذكر - أو الكلمات المحجوزة . ومن الكلمات التى يعرفها المبرمج مايلى :

- أسماء الملفات المستخدمة فى عبارات SELECT ووصف الملف FD ( انظر الاسطر رقم 32 ,33 ,39 ,51) من مثال ١ – ٢
- أسماء السجلات وأسماء الحقول داخل السجلات فى جزء البيانات (انظر الاسطر من رقم 43 الى رقم 49 ورقم 55 ومن رقم 59 الى رقم 68) .
- أسمأ المقاطع والأقسام فى جزء الاجراءات (انظر الاسطر رقم 72 ,80 ,87 ,93 ,105) .

**مثـال ٢ – ١٥ :**

( أ ) قارن أزواج الاسماء التى يعرفها المبرمج التالية :

- YTDAS or YEAR-TO-DATE-AVERAGE-SALES
- GPA or CUMULATIVE-GRADE-POINT-AVERAGE
- EXTRA or YEAR-END-BONUS-PAY

عندما يكون لديك شك.. اعد الهجاء ؛ فمن الصعب ان تقع فى الخطأ عندما تصف وصفا دقيقا .

( ب ) الاسماء التى يعرفها المستفيد التالية غير صحيحة او غير مرغوب فيها .

- RECORD ( غير صحيح : لاستخدامه كلمة محجوزة )
- STUDENT- BODY- CUMULATIVE- GRADE- POINT- AVERAGE
  ( غير صحيح : لانه اكثر من ٣٠ رمز )
- -OUT- OF- STOCK ( غير صحيح لانه بدء بالشرطة )
- N (غير مرغوب فيه : لانه لايصف شيئا)
- TOTAL- $ SALES ( غير صحيح : لانه يحتوى على رمز ممنوع استخدامه، وهو علامة الدولار )

## ٢ - ٥ البـــدء

عند هذه النقطة .. نكون قد خدشنا بالكاد سطح برمجة الكوبل . الا أن أى لغة برمجة يحسن تعلمها باستخدامها . ويخدم برنامجا الكوبل التاليان، كتوجيه فى حل تمارين البرمجة المعطاة فى نهاية هذا الفصل . ولاختبار البرامج التى تكتبها فعلا.. عليك باستشارة أستاذك او مبرمج خبير بالنسبة لعبارات تحكم العمل او عبارات لغة الأوامر اللازمة لجهاز الكمبيوتر المتاح لك.

**مثـال ٢ – ١٦ :**

برنامج كوبل كامل ؛ حيث إن كل التعليقات مقدمة كجزء من البرنامج ( باستخدام أسطر تعليق ) . تلى قائمة برنامج المصدر الفعلية هذه عينة للمخرجات .

```
       IDENTIFICATION DIVISION.

       PROGRAM-ID. PHONELST.

       AUTHOR.  LARRY NEWCOMER.
       INSTALLATION.  PENN STATE UNIVERSITY--YORK CAMPUS.

       DATE-WRITTEN.  MAY 1983.
       DATE-COMPILED. MAY  9,1983.
      *    PHONELST PRODUCES A PRINTED LISTING OF ALL EMPLOYEE
      *    NAMES AND PHONE NUMBERS.  NAMES AND PHONE NUMBERS ARE
      *    INPUT FROM A CARD FILE WHICH IS MAINTAINED IN ALPHABETICAL
      *    ORDER BY EMPLOYEE NAME.  THE ONLY OUTPUT IS A PRINTED
      *    COPY OF NAMES AND PHONE NUMBERS -- SINGLE SPACED, WITH
      *    NO HEADINGS, PAGE NUMBERS, ETC. (KEEP IT SIMPLE)

       ENVIRONMENT DIVISION.

       CONFIGURATION SECTION.

      *    THE CONFIGURATION SECTION DESCRIBES THE COMPUTER SYSTEM
      *    TO BE USED TO COMPILE THE COBOL PROGRAM
      *    (SOURCE-COMPUTER), AND THE COMPUTER SYSTEM TO BE USED TO
      *    EXECUTE THE RESULTING OBJECT PROGRAM (OBJECT-COMPUTER).

       SOURCE-COMPUTER.  IBM-370.
       OBJECT-COMPUTER.  IBM-370.

       INPUT-OUTPUT SECTION.

       FILE-CONTROL.

      *    THE FILE-CONTROL PARAGRAPH OF THE INPUT-OUTPUT SECTION
      *    BEGINS TO DEFINE THE FILES TO BE PROCESSED BY THIS PROGRAM:

           SELECT CARD-FILE              ASSIGN TO CARDS.
           SELECT PRINT-FILE             ASSIGN TO PRINTER.

      *    THESE STATEMENTS GIVE COBOL NAMES TO THE FILES TO BE
      *    PROCESSED.  IN THIS CASE THE INPUT FILE IS KNOWN AS
      *    CARD-FILE AND THE OUTPUT FILE IS KNOWN AS PRINT-FILE.

      *    THE SELECT STATEMENTS ALSO ASSIGN EACH FILE TO A JOB
      *    CONTROL LANGUAGE STATEMENT WHICH WILL PROVIDE FURTHER
      *    INFORMATION REGARDING THE ACTUAL I/O DEVICE TO BE USED.
      *    CONSULT YOUR INSTRUCTOR OR AN EXPERIENCED PROGRAMMER
      *    REGARDING WHAT JOB CONTROL LANGUAGE IS NEEDED TO RUN
      *    A COBOL PROGRAM ON YOUR SYSTEM.

       DATA DIVISION.

      *        THE DATA DIVISION DESCRIBES ALL DATA ITEMS TO BE
      *        PROCESSED BY THIS PROGRAM.

       FILE SECTION.

      *        THE FILE SECTION PROVIDES ADDITIONAL INFORMATION FOR
      *        EACH FILE NAMED IN A SELECT STATEMENT.  NOTICE THAT
      *        THE SELECT FILE NAME AND THE "FD" FILE NAME ARE THE
      *        SAME.  THERE MUST BE A FILE DESCRIPTION ("FD") FOR
      *        EACH FILE.
```

```
      FD  CARD-FILE
          RECORD CONTAINS 80 CHARACTERS
          LABEL RECORDS ARE OMITTED

     *        THE "FD" INDICATES THE NUMBER OF CHARACTERS IN A RECORD
     *        AND WHETHER OR NOT THE FILE HAS FILE LABELS.  FOR A
     *        STANDARD SIZE CARD, THE LENGTH IS 80 COLUMNS.
     *        FILE LABELS ARE NOT PERMITTED FOR CARD FILES.

     *        THE REST OF THE "FD" DESCRIBES THE RECORD CONTENTS.
     *        "PIC X(80)" MEANS EACH RECORD IS 80 ALPHANUMERIC
     *        CHARACTERS.


      01  NAME-PHONE-INPUT             PIC X(80).

     *        THE "FD" FOR A PRINT FILE MUST INDICATE NO FILE LABELS.
     *        MOST PRINTERS PRINT UP TO 132 CHARACTER LINES.

      FD  PRINT-FILE
          RECORD CONTAINS 132 CHARACTERS
          LABEL RECORDS ARE OMITTED


     *        A "RECORD" FOR A PRINT FILE IS A PRINTED LINE.  IN
     *        THIS CASE EACH LINE CONSISTS OF 132 ALPHANUMERIC
     *        CHARACTERS ("PIC X(132)").

      01  PRINT-LINE                   PIC X(132).

      WORKING-STORAGE SECTION.

     *        WORKING-STORAGE AREAS ARE NOT PART OF FILE RECORDS.
     *        HERE WE DEFINE A PROGRAM SWITCH WHICH CAN HOLD 3
     *        ALPHANUMERIC CHARACTERS ("PIC X(3)").  THE SWITCH
     *        CONTAINS THE VALUE "NO " AS LONG AS THERE ARE MORE
     *        INPUT RECORDS TO BE PROCESSED.  WHEN THERE ARE NO
     *        MORE INPUT RECORDS LEFT, THE SWITCH IS SET TO "YES".

      01  END-OF-CARDS-SWITCH          PIC X(3).

     *        THIS AREA HOLDS AN INPUT CARD WITH AN EMPLOYEE'S
     *        NAME AND PHONE NUMBER -- NAME IS 20 ALPHANUMERIC
     *        CHARACTERS; PHONE IS 8 ALPHANUMERIC CHARACTERS.
     *        "FILLER" IS USED TO DEFINE ANY PART OF A RECORD WHICH
     *        WILL NOT BE PROCESSED BY THE PROGRAM.

      01  NAME-PHONE-CARD.
          05  EMPLOYEE-NAME            PIC X(20).
          05  EMPLOYEE-PHONE-NUMBER    PIC X(8).
          05  FILLER                   PIC X(52).

     *     THIS AREA IS WHERE WE PUT TOGETHER A LINE TO BE PRINTED.
     *        NOTICE THERE ARE SOME AREAS BETWEEN DATA ITEMS WHICH
     *        HAVE BEEN NAMED "FILLER" AND SET TO BLANK SPACES
     *        WITH "VALUE SPACES".  THESE AREAS SEPARATE FIELDS
     *        ON THE PRINTED LINE.

      01  NAME-PHONE-LINE.
          05  PRINT-NAME               PIC X(20).
          05  FILLER                   PIC X(5)      VALUE SPACES.
          05  PRINT-PHONE-NUMBER       PIC X(8).
          05  FILLER                   PIC X(99)     VALUE SPACES.
```

```
     PROCEDURE DIVISION.

    *        THE PROCEDURE DIVISION CONTAINS THE INSTRUCTIONS
    *        WHICH TELL THE COMPUTER WHAT TO DO.

     000-PRODUCE-PHONE-LISTING.

         MOVE "NO " TO END-OF-CARDS-SWITCH

    *        SETS END-OF-CARDS-SWITCH TO "NO".

    *        THE OPEN STATEMENT MUST BE USED BEFORE A FILE CAN
    *        BE PROCESSED IN ANY WAY.  IT TELLS WHETHER THE FILE
    *        IS TO BE USED FOR INPUT OR OUTPUT.

         OPEN    INPUT                CARD-FILE
                 OUTPUT               PRINT-FILE

         PERFORM 100-INPUT-A-RECORD

    *            THE PERFORM STATEMENT CAUSES ALL STATEMENTS
    *            IN THE PARAGRAPH NAMED "100-INPUT-A-RECORD" TO BE
    *            EXECUTED. NOTE THAT THIS CAUSES THE FIRST RECORD
    *            OF THE FILE TO BE INPUT.  THE "PRODUCE-NAME-PHONE-
    *            LISTING" PARAGRAPH EXECUTED NEXT BEGINS BY
    *            PROCESSING THIS FIRST RECORD.  WHEN IT IS DONE
    *            PROCESSING THE RECORD, IT THEN PERFORMS THE
    *            100-INPUT-A-RECORD PARAGRAPH TO OBTAIN THE NEXT
    *            RECORD IN THE FILE.



         PERFORM 200-PRODUCE-NAME-PHONE-LIST
             UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"

    *                THIS VERSION OF PERFORM CAUSES THE ENTIRE
    *                PARAGRAPH NAMED "200-PRODUCE-NAME-PHONE-LIST"
    *                TO BE EXECUTED REPEATEDLY.  THE REPEATED
    *                INVOKING OF THE PARAGRAPH CONTINUES UNTIL
    *                END-OF-CARDS-SWITCH CONTAINS THE VALUE "YES".


         CLOSE   CARD-FILE
                 PRINT-FILE

    *            THE CLOSE STATEMENT MUST BE USED WHEN THE PROGRAM
    *            IS DONE PROCESSING A FILE.  IT "DISCONNECTS" THE
    *            FILE FROM THE COBOL PROGRAM.

         STOP RUN

    *            THE STOP STATEMENT TELLS THE COMPUTER TO STOP
    *            EXECUTING INSTRUCTIONS IN THIS PROGRAM.


     100-INPUT-A-RECORD.

    *                THIS PARAGRAPH INPUTS A CARD.   THE READ
    *                STATEMENT INPUTS THE NEXT CARD IN THE FILE AND
    *                MOVES THE CONTENTS OF THE CARD INTO THE
    *                WORKING-STORAGE AREA NAMED "NAME-PHONE-CARD".
    *                IF THERE ARE NO MORE CARDS TO BE READ
```

```
          *               ("AT END"), END-OF-CARDS-SWITCH IS SET
          *               TO "YES".  THIS CAUSES THE CLOSE AND STOP
          *               RUN STATEMENTS TO BE EXECUTED.

               READ CARD-FILE RECORD
                   INTO NAME-PHONE-CARD
                   AT END
                       MOVE "YES" TO END-OF-CARDS-SWITCH


           200-PRODUCE-NAME-PHONE-LIST.

          *        THIS PARAGRAPH MOVES THE DATA FROM NAME-PHONE-CARD
          *        TO THE AREA FROM WHICH A LINE OF THE REPORT WILL BE
          *        PRINTED.  THE "WRITE" STATEMENT PRINTS A LINE WHOSE
          *        CONTENTS ARE IN "NAME-PHONE-LINE".  HAVING PROCESSED
          *        ONE EMPLOYEE'S INFORMATION, "PERFORM 100-INPUT-A-RECORD'
          *        CAUSES THE READ STATEMENT IN PARAGRAPH 100-INPUT-A-RECOI
          *        TO BE EXECUTED, BRINGING THE NEXT RECORD INTO MEMORY
          *        FOR PROCESSING.

               MOVE EMPLOYEE-NAME            TO PRINT-NAME
               MOVE EMPLOYEE-PHONE-NUMBER    TO PRINT-PHONE-NUMBER
                .
               WRITE PRINT-LINE
                   FROM NAME-PHONE-LINE

               PERFORM 100-INPUT-A-RECORD
               .
```

```
ABEL FRED                123-4567
BAKER SUE                111-2222
CHARLIE SUE              453-7822
PERELMAN BARNEY          UNLISTED
PERELMAN MIKE            UNKNOWN
RODGERS BOB              234-5678
RODGERS BOBB             345-6789
RODGERS LEE              456-7890
FOLKERS DICK             567-8901
FOLKERS RUTH             678-9012
SUE PEGGY                231-7856
ZOTZ ZAPPA               999-9999
```

**مثـــال ٢ - ١٧**

يشبه مثال ٢ - ١٦ ، ونعلق على السمات الجديدة فقط .

```
       IDENTIFICATION DIVISION.

       PROGRAM-ID. TIMELIST.

       AUTHOR.  LARRY NEWCOMER.
       INSTALLATION.  PENN STATE UNIVERSITY--YORK CAMPUS.

       DATE-WRITTEN.  MAY 1983.
       DATE-COMPILED. MAY  9,1983.
      *    TIMELIST PRINTS ONE LINE FOR EACH EMPLOYEE SHOWING:
      *    NUMBER OF REGULAR HOURS WORKED, NUMBER OF OVERTIME
      *    HOURS WORKED, AND TOTAL HOURS WORKED. AT THE END OF THE
      *    REPORT, IT ALSO PRINTS A COUNT OF THE NUMBER OF
      *    EMPLOYEES PROCESSED.  INPUT IS FROM A DECK OF TIME
      *    CARDS, KEPT IN SEQUENCE BY EMPLOYEE NAME.

       ENVIRONMENT DIVISION.

       CONFIGURATION SECTION.

       SOURCE-COMPUTER. IBM-370.
       OBJECT-COMPUTER. IBM-370.

       INPUT-OUTPUT SECTION.

       FILE-CONTROL.

           SELECT CARD-FILE             ASSIGN TO CARDS.
           SELECT PRINT-FILE            ASSIGN TO PRINTER.

       DATA DIVISION.

       FILE SECTION.

       FD  CARD-FILE
           RECORD CONTAINS 80 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .

       01  TIME-CARD-INPUT              PIC X(80).

       FD  PRINT-FILE
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .

       01  PRINT-LINE                   PIC X(132).

       WORKING-STORAGE SECTION.

      *    OBSERVE THAT THERE ARE SEVERAL NUMERIC FIELDS DEFINED
      *    IN WORKING-STORAGE USING "PIC 99" OR "PIC 999".  EACH
      *    "9" IN THE DESCRIPTION OF A NUMERIC ITEM REPRESENTS
      *    ONE DECIMAL DIGIT POSITION. THUS "PIC 99" DESCRIBES
      *    A DATA ITEM WHICH CAN HOLD A 2-DIGIT NUMBER.

       01  END-OF-CARDS-SWITCH          PIC X(3).

       01  WORKING-TIME-CARD.

      *    HOLDS INFORMATION INPUT FROM THE TIME CARD FILE
```

```
           05  WORKING-NAME               PIC X(20).
           05  WORKING-REGULAR-HOURS      PIC 99.
           05  WORKING-OVERTIME-HOURS     PIC 99.
           05  FILLER                     PIC X(56).

       01  TIME-LISTING-LINE.

      *    AREA USED TO CONSTRUCT A LINE TO BE PRINTED ON THE REPORT

           05  LISTING-NAME               PIC X(20).
           05  FILLER                     PIC X(5)      VALUE SPACES.
           05  LISTING-REGULAR-HOURS      PIC 99.
           05  FILLER                     PIC X(5)      VALUE SPACES.
           05  LISTING-OVERTIME-HOURS     PIC 99.
           05  FILLER                     PIC X(5)      VALUE SPACES.
           05  LISTING-TOTAL-HOURS        PIC 999.
           05  FILLER                     PIC X(90)     VALUE SPACES.

       01  EMPLOYEE-TOTAL-LINE.

      *    AREA USED TO CONSTRUCT THE LINE TO BE PRINTED AT END OF
      *    THE REPORT.  CONTAINS COUNTER FOR NUMBER OF EMPLOYEES
      *    PROCESSED.

           05  FILLER                     PIC X(25)
                                          VALUE "NUMBER OF EMPLOYEES IS".
           05  NUMBER-OF-EMPLOYEES        PIC 999.
           05  FILLER                     PIC X(104)    VALUE SPACES.

       PROCEDURE DIVISION.

       000-PRODUCE-HOURS-REPORT.

      *            PROGRAM SWITCHES AND COUNTERS MUST BE INITIALIZED
      *            AT THE START OF PROGRAM EXECUTION.  THE SWITCH IS
      *            SET TO "NO " WHILE THE COUNTER FOR NUMBER OF
      *            EMPLOYEES IS INITIALLY SET TO ZERO.  ONE WILL BE
      *            ADDED EACH TIME AN EMPLOYEE IS PROCESSED (SEE
      *            200-PRODUCE-TIME-LISTING PARAGRAPH).

           MOVE "NO " TO END-OF-CARDS-SWITCH
           MOVE ZERO TO NUMBER-OF-EMPLOYEES

           OPEN    INPUT              CARD-FILE
                   OUTPUT             PRINT-FILE

           PERFORM 100-INPUT-A-RECORD

           PERFORM 200-PRODUCE-TIME-LISTING
               UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"

      *            AFTER ALL INPUT CARDS HAVE BEEN PROCESSED (I.E.,
      *            WHEN END-OF-CARDS-SWITCH HAS BEEN CHANGED TO
      *            "YES"), A LINE CONTAINING A COUNT OF ALL
      *            EMPLOYEES PROCESSED MUST BE PRINTED.

      *            "WRITE PRINT-LINE FROM EMPLOYEE-TOTAL-LINE" PRINTS
      *            THE CONTENTS OF EMPLOYEE-TOTAL-LINE.

           WRITE PRINT-LINE
               FROM EMPLOYEE-TOTAL-LINE
```

```

           CLOSE    CARD-FILE
                    PRINT-FILE
           STOP RUN
           .

       100-INPUT-A-RECORD.

           READ CARD-FILE RECORD
               INTO WORKING-TIME-CARD
               AT END
                   MOVE "YES" TO END-OF-CARDS-SWITCH
           .

       200-PRODUCE-TIME-LISTING.

      *            INCREMENT COUNTER EACH TIME AN EMPLOYEE IS PROCESSED

           ADD 1 TO NUMBER-OF-EMPLOYEES

      *            MOVE DATA FROM INPUT CARD TO AREA WHERE LINE IS BUILT

           MOVE WORKING-NAME           TO LISTING-NAME
           MOVE WORKING-REGULAR-HOURS  TO LISTING-REGULAR-HOURS
           MOVE WORKING-OVERTIME-HOURS TO LISTING-OVERTIME-HOURS

      *            CALCULATE TOTAL HOURS BY ADDING REGULAR, OVERTIME.
      *            NOTE THAT RESULT IS PLACED IN TIME-LISTING-LINE.

           ADD WORKING-REGULAR-HOURS  WORKING-OVERTIME-HOURS
               GIVING LISTING-TOTAL-HOURS

           WRITE PRINT-LINE
               FROM TIME-LISTING-LINE

           PERFORM 100-INPUT-A-RECORD

```

```
ABEL FRED                 40     03     043
BAKER SUE                 30     00     030
CHARLIE CHUCK             40     12     052
FLECKSTEINER CINDY        40     00     040
FLECKSTEINER DON          40     20     060
FLECKSTEINER STACY        15     00     015
PERELMAN BARNEY           40     15     055
PERELMAN MIKE             03     00     003
NUMBER OF EMPLOYEES IS    008
```

## أسئلة مراجعة

٢ – ١ ماذا يعنى القول ان لغة الكوبل هى لغة توثيق ذاتى ؟

٢ – ٢ ما الأجزاء الأربعة لبرنامج الكوبل ؟

٢ – ٣ ما الغرض العام من جزء التعريف ؟

٢ – ٤ ما الغرض العام من جزء الاوساط ؟

٢ – ٥ ما الغرض العام من جزء البيانات ؟

٢ – ٦ ما مخزن العمل ؟

٢ – ٧ ما الجزء الذى يحدد التشغيل الفعلى المراد أداؤه ؟

٢ – ٨ ما القسم فى برنامج الكوبل ؟

٢ – ٩ ما مقاطع برنامج الكوبل ؟

٢ – ١٠ لماذا يقترح أن تحتوى أسماء مقاطع جزء الاجراءات على ارقام متتالية كجزء من الاسم ؟

٢ – ١١ ما النقطة المرتبة ؟

٢ – ١٢ وضح المنطقة A والمنطقة B فى الكوبل .

٢ – ١٣ ما العناصر التى يجب ان تبدأ كتابتها فى المنطقة A ؟

٢ – ١٤ وضح اهمية الترحيل داخل المنطقة B.

٢ – ١٥ وضح قواعد الاستمرارية فى الكوبل .

٢ – ١٦ وضح قواعد كتابة برنامج الكوبل .

٢ – ١٧ ما الاستخدامات الخاصة للعمود السابع فى الكوبل ؟

٢ – ١٨ ما كلمات الكوبل المحجوزة ؟

٢ – ١٩ ما قواعد كتابة الاسماء التى يعرفها المبرمج ؟

٢ – ٢٠ وضح وظيفة الأسطر الفارغة فى برنامج الكوبل .

## مسائل محلولة

### ٢ – ٢١ ما أنواع عناصر البيانات التى تنتمى الى مخزن العمل ؟

يستخدم **مخزن العمل** لعناصر البيانات التى لاتكون حقولا فى سجلات ملفات . عادة ماتحسب مثل هذه العناصر بواسطة البرنامج ، او تستخدم كمناطق عمل مؤقتة اثناء التشغيل . وعادة ما يتحدد فى مخزن العمل مايلى :

- **عدادات** ، او مواقع ذاكرة تستخدم فى حفظ بيانات عددية تمثل عدد ( مثل عدد العاملين ) . عادة ماتوضع قيم ابتدائية مساوية للعدادات .

- **مركمات** ، او مواقع ذاكرة تستخدم فى تركيم اجمالى ( مثل اجمالى الموازنة المستحقة ) . وعادة ماتوضع قيمة مساوية صفر للمركمات accumulators.

- **اشارات و مفاتيح للبرنامج**، وهذه هى مواقع ذاكرة تحفظ معلومات، تحدد ما اذا كان احد الانشطة قد حدث اثناء تنفيذ البرنامج . عادة .. مايكون للاشارات flags والمفاتيح switches قيم : صحيح true او خطأ false ( مثل : اذا كانت كل سجلات الملف اجرى عليها التشغيل ام لا ) وتعد الاشارات ويعاد اعدادها كلما كانت هناك حاجة اثناء تنفيذ البرنامج .

• **مناطق صور الاسطر** . من المريح عادة عمل سطر معلومات فى مخزن العمل، ثم نقل هذا السطر الى مناطق سجل الملف طباعة قبل طباعته النهائية بعبارة write .

٢ – ٢٢ ماذا يفعل الكمبيوتر عندما ينفذ صيغة لغة الآلة من PERFORM 090- CALCULATE- TAX ؟

يتجه الكمبيوتر الى اول التعليمات فى صيغة لغة الآلة للمقطع المسمى 090- CALCULATE- TAX . ويستمر عند ذلك فى تنفيذ كل التعليمات الاخرى الموجودة فى المقطع ، واحدة تلو الاخرى ، واخيرا .. يعود الى التعليمة التى تلى PER- FORM 090- CALCULATE- TAX مباشرة .

٢ – ٢٣ ماذا يفعل الكمبيوتر عندما ينفذ صيغة لغة الآلة من MOVE INVENTORY- ON- HAND- AMOUNT TO IN- VENTORY- REPORT- AMOUNT ؟

في كلمات عامة ينسخ الكمبيوتر محتويات موقع الذاكرة ( الحقل الراسل ) INVENTORY- ON- HAND- AMOUNT فى موقع آخر للذاكرة ( الحقل المستقبل ) INVENTORY- REPORT- AMOUNT . وتظل قيمة INVENTORY–ON- HAND- AMOUNT دون تغيير . وتعتمد المحتويات الدقيقة للحقل المستقبل على وصف جزء البيانات للعنصرين .

٢ – ٢٤ حدد ما اذا كانت الاسماء التالية غير صحيحة او صحيحة لكنها ضعيفة بالنسبة الى اسلوب البرمجة او انها مقبولة بالكامل .

( أ ) INPUT

( ب ) INPUT- INVENTORY- RECORD

( جـ ) B- INPUT- INVENTOY- RECORD- 040

( د ) PROCESS- DATA

( أ ) غير صحيح : اسم يعرفه المستفيد ، يكرر كلمة محجوزة .

( ب ) صحيح ( يمكن تحسينها باضافة رقم متتالى للاسم ) .

( جـ ) صحيح : خليط الحروف والارقام المتتالية مع اسم يصف مايفعله المقطع ، يكون اختياريا .

( د ) صحيح : الا ان الاسم ضعيف ؛ حيث انه لايصف مايفعله المقطع بالفعل .

٢ – ٢٥ وضح الاستخدامات الخاصة التالية للعمود السابع.

```
(a)            MOVE SPACES TO WORKING-NAME-ADDR
          -        ESS-LINE
(b)       *    THE FOLLOWING PARAGRAPH COMPUTES STATE TAX
(c)            MOVE "THE PENNSYLVANIA STATE UNIVERSITY BO←col. 72
          -        "OKSTORE" TO REPORT-HEADING
(d)       D    EXHIBIT NAMED NUMBER-OF-EMPLOYEES-PROCESSED
          ↑    ↑
       col. 7  B Margin
```

( أ ) حيث ان WORKING- NAME- ADDRESS- LINE مقسم على سطرين.. فستكون هناك حاجة الى وجود شرطة فى العمود السابع؛ لتحديد الاستمرار . لاحظ امكانية تجنب الاستمرار على النحو التالى .

```
MOVE SPACES TO
    WORKING-NAME-ADDRESS-LINE
```

( ب ) النجمة الموجودة فى العمود السابع تحدد إن بقية السطر عبارة عن تعليق ، يستخدم لتوضيح مايفعله جزء معين من البرنامج وكيفية عمله ذلك

( جـ) تستخدم الشرطة الموجودة فى العمود السابع لاستمرار سلسلة موضوعة بين علامتى تنصيص (ثابت غير عددى) من سطر للسطر التالى له . لاحظ انه لاتوجد علامة تنصيص فى نهاية السطر الاول ( الذى ينتهى فى العمود السابع ) ، وانه توجد علامة تنصيص فى بداية المنطقة B من السطر الثاني . وقد كان من الممكن تجنب هذا الاستمرار بكتابة مايلى :

```
MOVE
    "THE PENNSYLVANIA STATE UNIVERSITY BOOKSTORE"
       TO REPORT-HEADING
```

( د ) تحدد D الموجودة فى العمود السابع ان السطر هو سطر تصحيح ، وتستخدم فى إيجاد اخطاء فى البرنامج . (انظر رقم ٢ من الفصل الرابع) .

٢ – ٢٦ علق على الاسماء التى يعرفها المبرمج لعناصر البيانات التالية :

(أ) TNWI

(ب) TOTAL- NET- WORTH- OF- INVENTORY- FOR- JULY

(جـ) PHONE NUMBER

(د) # - ON- HAND

(هـ) ON- HAND

( أ ) يجب ان يكون اسم البيانات وصفيا و بالرغم من ان TNWI صحيح الا ان TOTAL- NET- WORTH- IN-VENTORY أفضل .

( ب) يجب ألا يزيد طول الاسماء التى يعرفها المبرمج عن ٣٠ رمزا ، وعلى هذا فالاسم خطأ .

(جـ) يجب ألا توجد فراغات فى الأسماء التى يحددها المبرمج (كما يجب ألا تشتمل على أى رمز آخر سوى الحروف الابجدية من A الى Z ، والارقام من 0 الى 9 والشرطة فقط، وعلى هذا فالاسم خطأ.

( د ) غير صحيح لوجود الرمز الخاص # فى الاسم .

( هـ) لايمكن ان ينتهى الاسم بشرطة «غير صحيح ».

## مشـــاكل متكاملـــة

٢ – ٢٧ اعمل لقاءً مع مبرمج كوبل له خبرة فى البرمجة .

ناقش نمطيات كتابة الشفرة التى يستخدمها بالنسبة لـ ( أ ) اسطر التعليق. ( ب ) الاستمرارية . ( جـ ) الاسماء التى يعرفها المبرمج . ( د ) النقطة المرتبة . ( هـ ) الترحيل . ( و ) الاسطر الفارغة .

٢ – ٢٨ أعد قراءة مثال ٢ – ١ وانظر اذا كنت تستطيع ان تفهم معظم الموجود فى البرنامج ام لا . هل تستطيع ان تفهم البرنامج بدون التعليقات الموجودة فيه ؟ او بدون الاسطر الفارغة ؟ أو بدون الترحيل ؟ او بدون الاسماء الوصفية للبيانات؟

## تمـــارين برمجـــة

فيما يلى ... كل المبالغ النقدية معطاة مقربة بالدولار ( انظر الفصل الخامس لمعاملة الدولار والسنت ) .

٢ – ٢٩ اكتب برنامجا كاملا بلغة الكوبل ، واختبره وصححه لطباعة التقرير التالى . اختر بيانات اختيارية، وتحقق من صحة المخرجات .

١ – طول سطر التقرير 132 خانة، ويشمل مايلى:

على التوالى

customer number – ٥ خانات ، حرفى عددى

١٠ خانات فارغة

customer name – ٣٠ خانة ، حرفى عددى

١٠ خانات فارغة

item numbe – ٧ خانات ، حرفى عددى

٥ خانات فارغة

quantity ordered – ٤ خانات ، عددى

٥ خانات فارغة

quantity shipped – ٤ خانات ، عددى

بقية السطر فارغ

٢ – بيانات المدخلات مثقبة فى بطاقات على النحو التالى :

customer number – أول خمس اعمدة

customer name – الثلاثون عمود التالية

item number – السبع أعمدة التالية

quantity ordered – الأربع اعمدة التالية

quantity shipped – الأربع اعمدة التالية

بقية البطاقات – غير مستخدمة

٣ – أطبع فى نهاية التقرير سطرا يبين اجمالى عدد العناصر المطلوبة items ordered ( لكل العملاء ) ، واجمالى عدد العناصر المرسلة items shipped ( لكل العملاء )

٢ – ٣٠ أكتب برنامجا كاملا بلغة الكوبل واختبره وصححه لطباعة التقرير التالى :

| customer number | فراغ | old balance | فراغ | payment amount | فراغ | new balance |
|---|---|---|---|---|---|---|
| ٨ | ٥ | ٥ | ٥ | ٤ | ٥ | ٥ |
| خانات | خانات | خانات | خانات | خانات | خانات | خانات |
| حرفى عددى | | عددى | | عددى | | عددى |

والمدخلات مثقبة على بطاقات على النحو التالى :

| بقية الخانات (٦٣ خانة) غير مستخدمة | payment amount ٤ خانات | old balance ٥ خانات | customer number ٨ خانات |
|---|---|---|---|

تحسب الموازنة الجديدة new balance بطرح ماتم دفعه poyment من الموازنة القديمة old balance .

٢ - ٣١ أكتب برنامجا كاملا بلغة الكوبل، واختبره وصححه لطباعة التقرير التالى :

تحتوى المخرجات على الحقول التالية مع ترك مسافات مناسبة :

part number - 8 خانات ، حرفى عددى

on hand - 5 خانات ، عددى

on order - 5 خانات ، عددى

quantitiy - 6 خانات ، عددى

المدخلات مثقبة فى بطاقات على النحو التالى :

part number - أول 8 أعمدة

on hand- الخمس أعمدة التالية

on order - الخمس أعمدة التالية

بقية البطاقات فارغة

تحسب الكمية المتاحة quantity available بجمع الكمية الموجودة on hand على الكمية المطلوبة on order .

أطبع فى النهاية سطراً يبين إجمالى الكمية المطلوبة ( لكل الأجزاء ) .

٢ - ٣٢ أكتب برنامجاً كاملاً بلغة الكوبل واختبره وصححه لطباعة التقرير التالى :

المخرجات ( مع ترك المسافات المناسبة بينهما ) .

general ledger account number - ٤ خانات ، حرفى عدددى

description - ٢٠ خانة ، حرفى عددى

credit amount - ٥ خانات ، عددى

debit amount - ٥ خانات ، عددى

المدخلات مثقبة فى بطاقات ومرتبة : طبقا لرقم حساب دفتر الاستاذ العام general ledger account number ، وموجودة على النحو التالى :

general ledger account number - ٤ خانات

descriptien - الخانات العشرين التالية

credit amount - الخانات الخمس التالية

debit amount – الخانات الخمس التالية

بقية البطاقة – غير مستخدمة

أطبع فى النهاية سطرا ، يبين إجمالى الرصيد الدائن credit وإجمالى الرصيد المدين debit .

٢ – ٣٣ أكتب برنامجا بلغة الكوبل، واختبره وصححه لطباعة القائمة التالية من ملف العاملين الرئيسى :

تحتوى المخرجات على الحقول التالية مع وجود خمس خانات فارغة بين كل حقل وآخر :

employee ID – ٦ خانات ، حرفى عددى

number of deductions – خانتان ، عددى

last year's annual salary – ٥ خانات ، عددى

this year's annual salary المدخلات مثقبة فى بطاقات على النحو التالى :

employee ID – الاعمدة من رقم ١ الى رقم ٦

annual salary – ٥ خانات ، عددى

number of deductions – العمودان رقم ٧ – ٨

annual salary last year´s – الأعمدة من ٩ – ١٣

this year´s annual salary – الأعمدة من رقم ١٤ – ١٨

أطبع فى النهاية سطراً يبين إجمالى الرواتب لهذا العام ، والعام الماضى، وعدد العاملين الذى أجرى لهم تشغيل .

٢ – ٣٤ أكتب برنامجا بلغة الكوبل، واختبره وصححه لطباعة القائمة التالية من ملف حسابات الدائنين :

المخرجات (مع ترك ٨ خانات فارغة بين كل حقل وآخر) :

invoice number – ٤ خانات ، حرفى عددى

invoice date (mm yy dd) – ٦ خانات ، عددى

invoice amount – ٥ خانات ، عددى

discount amount – ٤ خانات ، عددى

amount due (less discount) – ٥ خانات ، عددى

المدخلات مثقبة فى بطاقات على النحو التالى

invoice number – الأعمدة من ١ – ٤

invoice amount – الأعمدة ٥ – ٩

discount amount – الأعمدة ١٠ – ١٣

invoice date (mm yy dd) – الأعمدة ١٤ – ١٨

أطبع فى النهاية سطرا ، يبين إجمالى قيمة الفاتورة، وإجمالى الخصومات وإجمالى القيمة مطروحا منه الخصومات .

٢ – ٣٥ أطبع تقريرا يبين الحقول التالية من ملف طلبة احد الجامعات :

Student # Name Year # Credits Completed

المدخلات مثقبة فى أعمدة بطاقات على النحو التالى :

الأعمدة ١ – ٩ – student number

الاعمدة ١٠ – ٢٩ – name

العمود رقم ٣٠ year (4, 3, 2, 1)

أطبع فى النهاية سطرا بإجمالى عدد الساعات التى أنهيت وإجمالى عدد الطلبة .

٢ – ٣٦ أطبع عناوين بريدية لها الشكل التالى :

```
M. MOUSE
1031 EDGECOMB AVE.
YORK, PA 17403
```

المدخلات مثقبة فى أعمدة بطاقات على النحو التالى :

الأعمدة ١ – ٢٠ – name

الأعمدة ٢١ – ٤٠ – street

الأعمدة ٤١ – ٥١ – city

الأعمدة ٥٢ – ٥٣ – state

الأعمدة ٥٤ – ٥٨ – zip code

**ملاحظة :** ( ١ ) أنت فى حاجة الى ثلاث مناطق تخيلية لسطر المخرجات ( المشكلة رقم ٢١ من هذا الفصل ) .

( ٢ ) كل سجل مدخلات ينتج عنه ثلاثة أسطر مخرجات.

( ٣ ) استخدم .

```
WRITE PRINT-LINE
    FROM WORKING-NAME-LINE
        AFTER ADVANCING 2 LINES
```

فى ترك سطر فارغ بين العناوين

٢ – ٣٧ أكتب برنامجًا لطباعة شيكات ، لها الشكل التالى :

```
                    ______________________
                    CHECK #

____________        ______________________
   PAYEE                AMOUNT (dollars)
```

المدخلات مثقبة فى بطاقات وتحتوى على اسم المستفيد من الشيك payee فى الأعمدة (١ - ٢)، وقيمة الشيك فى الأعمدة (٢١ - ٢٥) . ينتج البرنامج أرقام الشيكات بدءا برقم الشيك 100 # ورقم الشيك الذى يليه هو 110 ، ثم 120 .. وهكذا . بعد طباعة كل الشيكات ، أطبع سطرا يبين إجمالى المبالغ المدفوعة disbursed .

٢ - ٣٨ أكتب برنامجا يطبع تقريراً بنكيًا من مدخلات البطاقة التالية :

customer name – الأعمدة (١ - ٢٠)

saving account balance – الأعمدة (٢١ - ٢٥)

checking account balance – الأعمدة (٢٦ - ٣٠).

certificates of deposit balance – الأعمدة (٣١ - ٣٥)

يجب أن تحتوى المخرجات على سطرين لكل عميل يبينا اسم العميل والموازنات المختلفة له . وفى النهاية.. يطبع سطر يبين اجمالى موازنات المدخرات total saving واجمالى الشيكات total checkinjg واجمالى موازنات الشهادات total certificate sbalances . التخطيط الدقيق للتقرير متروك لك .

# الفصل الثالث

# جــــزء التعــــريف

# Identification Division

## ٣ - ١ ترميز التكوين

لايسرى هذا القسم على جزء التعريف فقط ، بل يسرى كذلك على بقية برنامج الكوبل أيضا .

تشير كلمة تكوين syntax إلى التكوين النحوى للغة . مثال ذلك ، الجملة الانجليزية "dog the chaxed cat the" بها عديد من الاخطاء التكوينية syntax errors تشمل الهجاء وترتيب الكلمات . إذا أحتوى برنامج كوبل اخطاء تكوينية ، فلن يستطيع المترجم ان يترجم الجملة ( الجمل ) الفردية ، ولن يتواجد برنامج تنفيذ لتنفيذه . وتساعد رسائل الخطأ التشخيصية ، والتى تطبع كجزء من قائمة برنامج المصدر ، المبرمج فى ايجاد الاخطاء التكوينية وتصحيحها .

وقد طور ترميز شائع الاستخدام لتوضيح التكوين الصحيح لجمل الكوبل ، وتستخدم فيه الاصطلاحات التالية .

( ١ ) الأقواس المربعة [ ] تحتوى على مجموعة من عنصر واحد او أكثر اختيارية optional . تستطيع أن تختار أى واحد من المجموعة أو لا تأخذ شيئا على الإطلاق .

( ٢ ) الأقواس { } تحتوى على مجموعة من عنصر واحد او اكثر . ويجب ان تختار واحداً بالضبط exactly one من هذه العناصر.

( ٣ ) حروف كبيرة Uppercase leters تستخدم للكلمات المحجوزة ، وتستخدم الحروف الصغيرة lowercase letter للكلمات التى يعرفها المبرمج .

( ٤ ) يجب أن تظهر الكلمات المحجوزة التى يوضع تحتها خط على حالها فى العنصر ، اما الكلمات المحجوزة الأخرى .. فيمكن حذفها .

( ٥ ) النقاط ( ... ) تستخدم لتعريف عناصر يمكن ان تتكرر اى عدد من المرات .

**مثــــال ٣ - ١**

اعتبر الترميز التالى لعبارة MOVE من جزء الإجراءات .

```
MOVE {identifier-1} TO identifier-2 [identifier-3] . . .
     {literal     }
```

يوضع خط تحت كلمة MOVE، ولا تظهر الكلمة بين قوسين مربعين ، وعلى هذا .. فلابد من ظهورها . ويستطيع المبرمج أن يستخدم أما معرفًا Identifier او ثابتًا literal ( الثوابت التى تعرضنا لها حتى الآن هى السلاسل الموجودة بين علامتى تنصيص فقط ) ؛ حيث إنها بالحروف الصغيرة فيقوم المبرمج بتحديد المعرف ( او الثابت ) المطلوب ، والذى يتبعه المعرف المطلوب ( الذي يحدده المبرمج ) . ويمكن ان يتبع هذا بدوره عددا ( يشمل لاشىء ) من المعرفات التى يعرفها المبرمج .

**مثـــال ٣ – ٢ :**

نعلق على تحقيقات عديدة لعبارة مثال ٣ – ١

- MOVE INPUT-CUSTOMER-NAME TO OUTPUT-CUSTOMER-NAME

صحيحة : فهى فى صيغة MOVE identifier-1 To identifier - 2

- MOVE "OVERPAID" TO INVOICE-MESSAGE-AREA TYPE-OF-ACCOUNT

صحيحة : فهى فى صيغة : MOVE literal To identifier 2 identiiifier- 3

- MOVE DISCOUNT-AMOUNT TO INVOICE-DISCOUNT AND SHIPPING-DISCOUNT

غير صحيحة : يجب ألا تظهر AND بين المعرفات ، SHIPPING- DISCOUNT, INVOICE- DISCOUNT

- MOVE GREETING TO "HELLO"

غير صحيحة : بالرغم من أن "HELLO" ثابت صحيح .. ألا أنه من الممنوع ظهور ثابت بعد كلمة TO .

- MOVE "HELLO" GROSS-SALES TO GREETING SALES-TOTAL

غير صحيحة : يظهر معرف واحد فقط ، أو ثابت واحد فقط بين MOVE و TO .

## ٣ - ٢ تكوين جزء التعريف

جزء التعريف الذى سبق ذكر عمله فى القسم الأول من الفصل الثانى – الشكل التكوينى المبين فى شكل ٣ – ١

IDENTIFICATION DIVISION.
PROGRAM-ID. program-name.
[AUTHOR. comment.]
[INSTALLATION. comment.]
[DATE-WRITTEN. comment.]
[DATE-COMPILED.]
[SECURITY. comment.]

شكل (٣ - ١)

لاحظ أن عنوان الجزء ( اسم الجزء ) ، ومقطع PROGRAM- ID هما الذى يجب ظهورهما فى هذا الجزء، أما بقية المقاطع فظهورها اختيارى .

**مثـــال ٣ - ٣ :**

قارن الاسطر من رقم ١ الى رقم ٨ فى مثال ٢ . ١ بالنموذج التكوينى الموجود فى شكل ٣ . ١ .

## ٣ - ٣ مقطع تعريف البرنامج

يستخدم مقطع تعريف البرنامج PROGRAM- ID لتحديد اسم لبرنامج الهدف . وعندما يوضع هذا البرنامج فى صورته النهائية فى ملف قرص ( فى مكتبة البرامج ) ... يعرف باسمه الموجود فى هذا المقطع ( والذى يحفظ فى دليل المكتبة ) ، وتحت هذا الاسم يمكن الاتصال بالبرنامج لتنفيذه دون تعقيدات .

يجب أن يتفق هذا الاسم بصفة عامة مع قواعد كتابة الاسماء التى يعرفها المبرمج ( القسم الخامس من الفصل الثانى ) . إلا أن مترجم الكوبل - فى نظم IBM OS/VS ونظم أخرى ، يحول تعريف البرنامج بالكوبل الى اسم من نوع اسماء برامج المكتبة ويحدث ذلك تلقائيا . وبالنسبة الى كوبل IBM OS/VS : ( ١ ) يجب الا يزيد الاسم عن ٨ خانات ، ( ٢ ) يجب الا يحتوى الا على حروف هجائية وارقام فقط ، ( ٣ ) يجب ان يكون اول رمز حرفا هجائيا .

**مثـــال ٣ - ٤**

- INVENTORY-REPORT سوف تصبح INVENTOR

( مسموح بثمان خانات فقط ، يحذف المترجم بقية الحروف).

- INVRPT لايحدث لها أي تغيير .
- PRINT-PAYCHECKS سوف تصبح PRINT 0 PA

(مسموح بثمان خانات فقط ، غير مسموح بالشرطة ، وعلى هذا يحولها المترجم إلى 0 ).

- 1- UPDATE- RECIEVABLE سوف تصبح A 0 UPDATE

( مسموح بثمان خانات فقط ، غير مسموح بالشرطة ، وعلى هذا يحولها المترجم إلى O ، غير مسموح بالبدء برقم وعلى هذا يستبدله المترجم بالحرف A ).

• RECVUPDT لايحدث أى تغيير .

## ٣ - ٤ مقاطـع اختيـارية

جميع المقاطع المتبقية فى جزء التعريف اختيارية ، وتعامل الجمل الموجودة فى كل مقطع منها على أنها تعليقات . ونظراً لأن هذه المقاطع قد تحتوى على أى شيء يريد المبرمج كتابته ، فيجب أن تعكس الاسماء نوع البيانات المقدمة بوضوح .

**مثـال ٣ - ٥ :**

• INSTALLATION. RESEARCH AND DEVELOPMENT.

يحدد مقطع INSTALLATION مكان مركز الكمبيوتر الذى أعد فيه البرنامج ( يمكن أن يكون بالمنشأة الكبيرة عديد من نظم الكمبيوتر ، المنتشرة فى مواقع جغرافية مختلفة ) .

• DATE-COMPILED.

مقطع DATE-COMPILED خاص لأن المترجم ، وليس المبرمج ، يتولى ملؤه .

• SECURITY. AUTHORIZED PERSONNEL ONLY--SEE POLICY MANUAL 301.

يحدد مقطع SECURITY أى اعتبارات أمن تسرى على هذا البرنامج أو على الملفات التي يقوم بتشغيلها .

باتباع المقاطع المختلفة لجزء التعريف.. يوصى - بشدة - أن يدخل المبرمج مجموعة من بطاقات الاوامر ( مع وجود نجمة فى العمود السابع ) معطيا وصفا موجزا لما يفعله البرنامج وهذا يوفر الوقت والمال عندما يأتى مبرمج آخر فيما بعد ليدخل تغييرات على هذا البرنامج ( وهذا يسمى ببرمجة صيانة maintenance programming ) . يهدف التوثيق الجيد للبرنامج إلى جعل البرنامج أسهل فى قراءته وفهمه وتعديله .

**مثـال ٣ - ٦ :**

انقد جزء التعريف التالى لبرنامج ينفذ على نظام كمبيوتر IBM OS/VS :

```
PROGRAM-ID. PRINT-MAILING-LABELS.
DATE-WRITTEN. OCT 1983.
```

تقنيا ... الشيء الوحيد الخطأ هو أن عنوان الجزء (IDENTI FICATION DIVISION) غير موجود إلا أنه :

( ١ ) LABELS - MAILING- PRINT ليست فى صورة مكتبة برامج ، ويحولها المترجم إلى PRINT 0 MA.

( ٢ ) حذف DATE- COMPILED, INSTALLATION, AUTOUR و SECURITY ، وعبارات التعليق معطيا عرضا موجزا لما يفعله البرنامج، بما يمثل برمجة ضعيفة . التوثيق الجيد هو جزء أساسى لاى برنامج كوبل .

## أسـئلة مراجعـة

٣ - ١ ما الغرض من جزء التعريف ؟

٣ - ٢ عرف ( أ ) تكوينى ( ب ) خطأ تكوينى ( جـ ) رسائل تشخيصية ؟

٣ - ٣ وضح ترميز التكوين المستخدم فى هذا الكتاب .

٣ - ٤ اذكر تكوين جزء التعريف .

٣ - ٥ ما الأجزاء التى يجب ظهورها فى جزء التعريف ؟ وما الأجزاء الاختيارية ؟

٣ - ٦ وضح قواعد كتابة جزء التعريف ؟

٣ - ٧ ما وظيفة مكتبة البرامج ؟

٣ - ٨ ما قواعد مقطع تعريف البرنامج فى نظم IBM ؟

٣ - ٩ وضح ما المعلومات التى تعطى فى كل من المقاطع الاختيارية من جزء التعريف .

٣ - ١٠ ماهى برمجة الصيانة ؟ كيف ترتبط بجزء التعريف ؟

## مسـائل محلـولة

٣ - ١١ بمعرفة النموذج التالى لعبارة PERFORM من جزء الإجراءات

```
PERFORM {paragraph-name-1} [{THROUGH} {paragraph-name-2}]
        {section-name-1  } [{THRU   } {section-name-2  }]

        {identifier-1} TIMES
        {integer-1   }
```

حدد ما إذا كان مايلى صحيحا تكوينيا، أم لا :

(*a*) PERFORM CALCULATE-DISCOUNT-AMOUNT 10 TIMES
(*b*) PERFORM READ-EMPLOYEE-RECORD THROUGH PRINT-PAYCHECKS NUMBER-OF-EMPLOYEES TIMES
(*c*) PERFORM CHECK-DEPENDENT-STATUS TO COMPUTE-DEDUCTIONS NUMBER-OF-DEPENDENTS TIMES
(*d*) PERFORM THRU VALIDATE-ZIP-CODE 9 TIMES
(*e*) PERFORM CHECK-DEPARTMENT-CODE THROUGH CODE-COUNT TIMES
(*f*) PERFORM PRINT-INVOICE-LINE 7

( أ ) صحيح

( ب ) صحيح

( جـ) غير صحيح : يجب استخدام THROUGH أو THRU وليس TO

( د ) غير صحيح : أى من : section- name-1 أو paragraph- name-1 مطلوب :

PERFORM VALIDATE- NAME THRU VALIDATE- ZIP- CODE 9 TIMES .

( هـ) غير صحيح : أي من paraagraph- name-2 او section- name-2

يجب أن يتبع THROUGH أو THRU :

PERFORM CHECK- DEPARTMENT- CODE THRU PRINT- DEPARTMENT-
.TOTAL CODE- COUNT TIME.S

( و ) غير صحيح : TIMES مطلوبة :

. PERFORM PRINT- INVOICE- LINE 7 TIMES

٣ - ١٢ أى مما يلى يعتبر تعريف برنامج مناسب :

(*a*) MONTHLY-SALES-REPORT
(*b*) PRINT-ACCOUNTS-RECEIVABLE-AGED-TRIAL-BALANCE

( أ ) صحيح : الاسم وصفى بطريقة مناسبة لما يفعله البرنامج . يحول مترجم كوبل IBM الاسم الى MONTHLY 0، وعلى هذا يكون من الأفضل استخدام اختصار من ٨ خانات فى المواقع الأولى مثل SALESRPT .

( ب ) غير صحيح : يجب أن يكون تعريف البرنامج اسما يعرفه المبرمج ( محددا بعدد ٣٠ خانة ) .

٣ - ١٣ أكتب جزء التعريف من برنامج كوبل لطباعة شيكات . كل المعلومات الخاصة بالشيكات موجودة فى ملف قرص ، وسبق إنتاجها ببرنامج رواتب آخر .

فى الحل التالى .. لاحظ استخدام أسطر فارغة لتوضيح تعريف البرنامج .

```
IDENTIFICATION DIVISION.

PROGRAM-ID. PRNTCHCK.

AUTHOR. MIKE PERELMAN, JUNIOR PROGRAMMER, PAYROLL APPLICATIONS.
INSTALLATION. XYZ COMPANY, BALTIMORE, MD.
DATE-WRITTEN. 7 SEPT., 1983.
DATE-COMPILED.
SECURITY. SEE POLICY MANUAL 7A3 REGARDING HANDLING OF
          COMPANY CHECKS.

* PRNTCHCK PRINTS EMPLOYEE PAYCHECKS BY COPYING THE PAYROLL
* INFORMATION ALREADY ON DISK FILE PAYCHECK.IMAGE ONTO
* PRE-PRINTED CHECKS IN THE PRINTER. THE OPERATOR HAS
* THE CAPABILITY TO STOP OR BACK-UP PRINTING IN CASE OF
* DIFFICULTY WITH THE FORMS AND/OR PRINTER. PRNTCHCK ALSO
* LOGS THE FOLLOWING CONTROL INFORMATION: # CHECKS PRINTED,
* # CHECKS BACKED-UP, TOTAL $ AMOUNT OF CHECKS.
```

## تمارين برمجة

من ٣ – ١٤ إلى ٣ – ٢٢ عدل التمارين من رقم ٢ – ٢٩ / ٢ – ٣٨ فى الفصل الثانى، لتعكس ما تعلمته الآن عن جزء التعريف .

# الفصل الرابع

# جــزء الأوســاط

# Environment Division

## ٤ - ١ تكوين جزء الأوساط

لجزء الأوساط الذى ذكرت وظيفته العامة فى القسم الاول من الفصل الثانى، الشكل التكوينى المبين فى شكل (٤ – ١) (أدخلت ارقام الأسطر لتسهيل الاشارة إليها ) . ويمكن رؤية مكونات جزء الأوساط، وهى : قسم تشكيلCONFIGURATION SECTION ، وقسم مدخلات ومخرجات INPUT- OUTPUT SECTION . ويشمل القسم الأول مقاطع كمبيوتر المصدر SOURCE- COMPUTER وكمبيوتر الهدف OBJECT- COMPUTER ، وأسماء خاصة SPECIAL- NAMES . أما القسم الثانى.. فيشمل مقطعا واحدا فقط ، وهو التحكم فى الملفات FILE- CONTROL . تذكر ان عنوان الجزء ( السطر الأول من شكل ٤ . ١ )، وعناوين الأقسام ( السطران رقم ٢ ، ١٤) ، وعناوين المقاطع ( الاسطر : ٣ و ٤ و ٦ و ١٥ ) ، يجب أن تبدأ كتابة بقية محتوياتها فى المنطقة B . أحيانا تكتب محتويات المقطع على نفس السطر المكتوب فيه عنوان المقطع، وأحيانا لاتكتب. يجب أن يكون المظهر المرئى الجيد دليلك فى البرمجة .. ( انظر مثال ٤ – ١ ) . ولاتناقش هنا الا المحتويات التى تسرى على النظام التتابعى للملفات .

**مثـال ٤ – ١ :**

```
ENVIRONMENT DIVISION.

CONFIGURATION SECTION.

SOURCE-COMPUTER.  IBM-370 WITH DEBUGGING MODE.
OBJECT-COMPUTER.  IBM-370
                      PROGRAM COLLATING SEQUENCE IS
                        CUSTOMER-NUMBER-CODE-SEQUENCE.
SPECIAL-NAMES.

    C01 IS TOP-OF-PAGE
    CUSTOMER-NUMBER-CODE-SEQUENCE IS
         "3" THRU "9"
         "2"
```

```
                "0" THRU "1"


       INPUT-OUTPUT SECTION.

       FILE-CONTROL.

           SELECT CUSTOMER-SALES-FILE
               ASSIGN TO CUSTSALE
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
               FILE STATUS IS CUST-SALES-FILE-STATUS
               .

           SELECT SALES-REPORT
               ASSIGN TO SALERPT
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
               FILE STATUS IS SALES-REPORT-STATUS
               .
```

```
(1)  ENVIRONMENT DIVISION.
(2)  CONFIGURATION SECTION.
(3)  SOURCE-COMPUTER. computer-name [WITH DEBUGGING MODE] .
(4)  OBJECT-COMPUTER. computer-name
(5)      [PROGRAM COLLATING SEQUENCE IS alphabet-name] .
(6)  [SPECIAL-NAMES.
(7)      [function-name IS cobol-name] ...
(8)      [alphabet-name IS
(9)        { STANDARD-1
(10)         NATIVE
             literal-1 [{THROUGH} literal-2              ] ...
(11)                   [{THRU   }                        ]
                       [ ALSO literal-3 [ALSO literal-4] ]
            [literal-5 [{THROUGH} literal-6              ]] ...
(12)        [          [{THRU   }                        ]]
            [          [ ALSO literal-7 [ALSO literal-8] ]]  } ]
(13)  . ]
(14) [INPUT-OUTPUT SECTION.
(15) FILE-CONTROL.
(16)     SELECT file-name
(17)         ASSIGN TO external-name
(18)         [ORGANIZATION IS SEQUENTIAL]
(19)         [ACCESS MODE IS SEQUENTIAL]
(20)         [FILE STATUS IS data-name]
(21)         [RESERVE integer {AREAS}]
                              {AREA }
(22)
```

شكل (٤ - ١)

## ٤ - ٢ قسم التشكيل :

## مقطع كمبيوتر المصدر

يصف قسم التشكيل نظام الكمبيوتر المستخدم فى ترجمة البرنامج وتنفيذه . ويبدأ مقطع كمبيوتر المصدر SOURCE-COMPUTER باسم الكمبيوتر الذى يحدد الكمبيوتر المستخدم فى الترجمة . وتعامل محتويات المقطع كتعليق ولاتخدم إلا كتوثيق فقط للبرنامج . وتختلف صيغة اسم الكمبيوتر من نظام كمبيوتر لنظام آخر : وبالنسبة لنظم كمبيوتر IBM طراز 370 تكفى كتابة IBM- 370.

هناك محتوى اختيارى كذلك لمقطع كمبيوتر المصدر وهو WITH DEBUGGING MODE ( السطر الثالث فى شكل ٤ - ١ ) . فاذا كتب هذا الجزء تترجم كل أسطر التصحيح ( والمحدد لها الحرف D فى العمود السابع منها ) إلى برنامج الهدف وينفذها على ذلك الكمبيوتر أثناء تنفيذه برنامج الهدف . فاذا لم يوجد هذا الجزء.. تعامل أسطر التصحيح مثل التعليقات (أى إنه بالرغم من طباعتها فى قائمة برنامج المصدر، إلا أنها لا تترجم إلى برنامج الهدف وعلى هذا لايكون لها أى تأثير على التنفيذ) . ويعالج التصحيح بالكامل فى الفصل التاسع .

**مثــال ٤ - ٢ :**

- SOURCE-COMPUTER. IBM-370-158.

يضاف طراز الكمبيوتر رقم 158 اختياريا إلى IBM 370 وحيث إنه لم يتحدد WITH DEBUGGING MODE، تعامل العبارات الموجودة حرف D فى عمودها السابع كتعليقات، وعلى هذا لايكون لها أى تأثير على التنفيذ . ويعامل اسم الكمبيوتر كتعليق .

- SOURCE-COMPUTER. IBM-370
  WITH DEBUGGING MODE.

تترجم أسطر التصحيح الى برنامج الهدف ( مثل العبارات المعتادة ) وتنفذ . ويكتب جزء WITH DEBUGGING MODE فى سطر خاص به لتسهل رؤيته .

## ٤ - ٣ قسم التشكيل :

## مقطع كمبيوتر الهدف

يحدد مقطع كمبيوتر الهدف OBJECT- COMPUTER الكمبيوتر المستخدم فى تنفيذ برنامج الهدف . ويخدم اسم الكمبيوتر كتوثيق للبرنامج أساسا ، وعادة مايكون هو نفس اسم الكمبيوتر المستخدم فى مقطع كمبيوتر المصدر .

تذكر ان الكمبيوتر يخزن معلومات عن طريق تمثيل كل رمز حرفى عددى برقم ثنائى فريد . عندما تتراوح هذه الارقام من أقلها إلى أكبرها ... تنتج ترتيبا للرموز المناظرة ، يسمى تسلسل التتابع collating sequence. ويعتمد تسلسل التتابع ، بالطبع، على الشفرة الثنائية المستخدمة ، ويوضح تسلسل التتابع لكل من شفرة EBCDIC ، وشفرة ASCII فى ملحق ب .

**مثـال ٤ - ٣ :**

فى تسلسل التتابع لشفرة EBCDIC :

space < $ < # < A < B < Z < 0 < 9

نفس الرموز مرتبة فى تسلسل التتابع لشفرة ASCII على النحو التالى :

space < # < $ < 0 < 9 < A < B < Z

تحتفظ كل من شفرة EBCDIC ، وشفرة ASCII بنفس الترتيب الأبجدى المعتاد أى أن "ZIG" > "ZAG".

ويوجد جزء اختيارى كذلك فى مقطع كمبيوتر الهدف ، ( السطر الخامس من شكل ٤ - ١ ) والذى يسمح للمبرمج باختيار تسلسل تتابع لاستخدامه أثناء تنفيذ البرنامج . فإذا حذف هذا الجزء من البرنامج.. يسرى التسلسل التقليدى لنظام الكمبيوتر ( وحيث إن التسلسل التقليدى غالبا مايكون هو تسلسل التتابع المرغوب فيه .. فعادة مايحذف هذا الجزء ) . يحدد تسلسل التتابع النتائج عندما تقارن عناصر حرفية عددية، أو عندما تخزن هذه العناصر .

**مثـال ٤ - ٤ :**

ارجع الى مثال ٤ - ١ . فى مقطع كمبيوتر الهدف ( الاسطر من ٣٦ - ٣٨ ) يعامل IBM-370 كتعليق ، بينما يدعى CUSTOMER- NUMBER- CODE- SEQUENCE بتسلسل تتابع يستخدم فى البرنامج . وهذا التسلسل معرف بواسطة الاسطر من (٤٢ - ٤٥) من مقطع الأسماء الخاصة ليكون كما يلى :

3 < 4 < 5 < 6 < 7 < 8 < 9 < 2 < 0 < 1

## ٤ - ٤ قسم التشكيل :

## مقطع الأسماء الخاصة

لهذا المقطع ثلاث وظائف رئيسية يمكن أن تعالج مستقلة عن بعضها البعض .

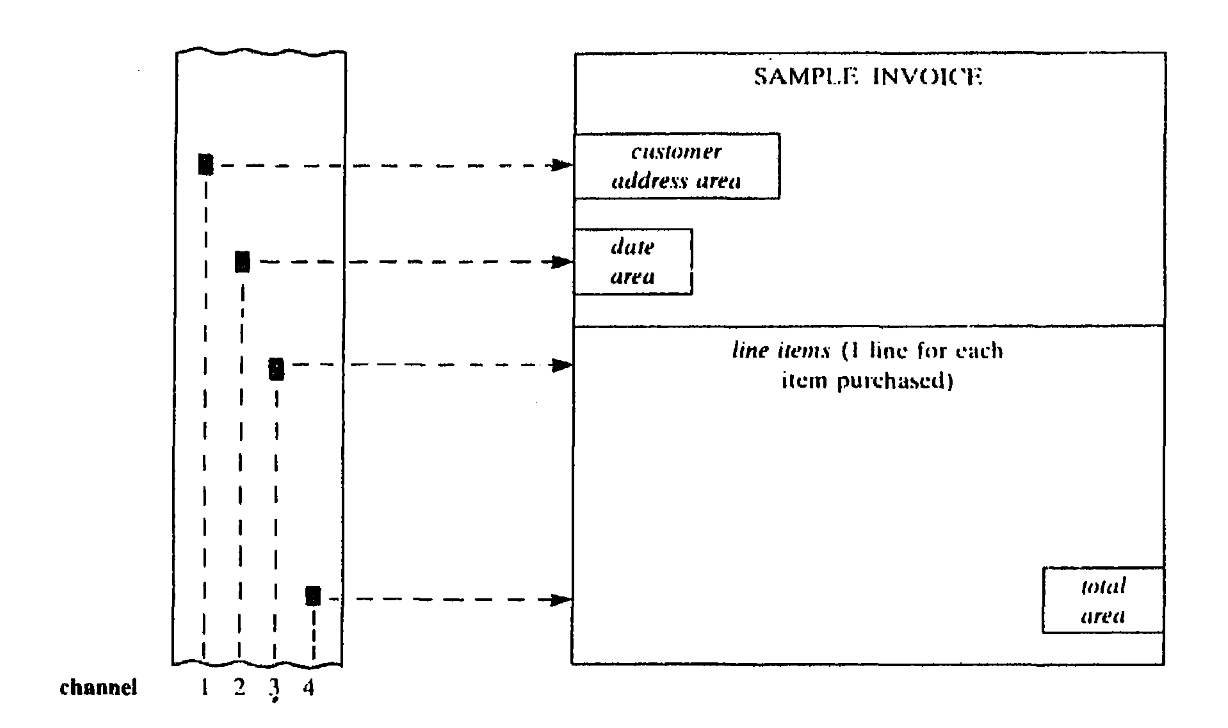
## تسمية قنوات تحكم العربة

عندما تطبع المخرجات على اوراق مستمرة، او صيغ مستمرة continuoed forms ... تعرف مواقع عناصر البيانات المختلفة فى الصيغة عن طريق قنوات تحكم العربة carriage control channels. وهذه القنوات محددة فى شكل ٤ - ٢ كثقوب فى شريط الورق الذى يتحرك مع الصيغة الاكثر إحتمالا، فى الطابعات الحديثة ... يمكن تحقيق أوضاع القناة إليكترونيا . وفى أى حالة من الحالات ، يستخدم السطر السابع من شكل ٤ - ١ فى تحديد اسم كوبل لقناة تحكم العربة للطابع . وفى كوبل IBM OS/VS يحتوى الاسم الوظيفى لقناة على الحرف C ، يليه رقميان كرقم للقناة : C01 , C02 و ... و... C12 . يمكن أن يكون أى اسم يضعه المبرمج اسم كوبل للقناة.

**مثـال ٤ – ٥ :**

```
C01 IS TOP-OF-INVOICE
C02 IS INVOICE-DATE-AREA
C03 IS CUSTOMER-ADDRESS-AREA
C04 IS LINE-ITEM-AREA.
```

على هذا .. فالقنوات من ١ الى ٤ تصاحب الاسماء التى يعرفها المبرمج، والتى تستخدم فيما بعد فى جزء الاجراءات للتحكم فى موقع المخرجات المطبوعة . لاحظ إنه لاتوجد الا نقطة واحدة فقط لكل مقطع الاسماء الخاصة ( السطر ١٣ من شكل ٤ – ١) – لاتستخدم نقاط بين المحتويات . لاحظ كذلك الرمز الوصفى للاسماء التى يضعها المبرمج .




شكل (٤ – ٢)

## تعريف تسلسل التتابع

إذا ما تحدد تسلسل تتابع غير تقليدى فى مقطع كمبيوتر الهدف ، فيجب أن يعرف تسلسل التتابع هذا فى مقطع الاسماء الخاصة ( الاسطر من رقم ٨ الى رقم ١٢ من شكل ٤ – ١ ). يجب أن يكون الاسم الأبجدى ( السطر الثامن ) فى مقطع الاسماء الخاصة مثل الاسم الموجود فى مقطع كمبيوتر الهدف ( السطر الخامس ) . وعلى هذا .. يعرف تسلسل التتابع المصاحب للاسم الابجدى كأحد الاشياء التالية :

- STANDARD-1

( السطر رقم ٩ من شكل ٤ – ١ ) يحدد تسلسل التتابع لشفرة ASCII

• NATIVE

( السطر رقم ١٠ من شكل ٤ – ١ ) يحدد تسلسل التتابع لشفرة EBCDIC ، وهى التقليدية لنظم كمبيوتر IBM من طراز 370 ) .

• اسم يعرفه المبرمج

( السطران رقم ١١ ، ١٢ من شكل ٤ – ١ ) يستطيع المبرمج أن يعرف تسلسل تتابع خاصاً به عن طريق كتابة سلسلة من الرموز بالإيجاز (literal-1 THROUGH literal-2) ، وتحدد تساوى رمزين بالبديل ALSO .

مرة أخرى نشير إلى إنه بغض النظر عن عدد قنوات تحكم العربة، وتسلسلات التتابع المعرفة فى مقطع الأسماء الخاصة .. فليست هناك سوى نقطة واحدة فقط ، في نهاية المقطع كله .

**مثـال ٤ – ٦ :**

```
OBJECT-COMPUTER. IBM-370 COLLATING SEQUENCE IS RUSH-ORDER-CODES.
SPECIAL-NAMES.

    C01 IS TOP-OF-FORM
    C02 IS MAILING-ADDRESS-AREA

    RUSH-ORDER-CODES IS

        "I" THRU "N"
        "D" "G" "F" "E"
        "A" THRU "C"
        "H" ALSO "O"
```

هنا تعرف الاسماء الخاصة تسلسل التتابع المكتسب RUSH- ORDER- CODE كما يلى :

$$I < J < K < L < M < N < D < G < F < E < A < B < C < H = O$$

تعتبر H و O متساويتان فى التتابع. يفترض فى أى رموز غير محددة انها تأتى فى النهاية (بعد H و O) وتكون فى نفس ترتيبها الطبيعى، أى أن P تتبع H و O.

## تسمية SYSOUT, SYSIN ، والشاشة المرئية للمشغل الآدمى

يمكن أن يحدد مقطع الاسماء الخاصة اختياريا ( السطر السابع من شكل ٤ – ١ ) أسماء كوبل للفات نظام متخصص معين ، يتم تشغيلها بعبارات DISPLAY, ACCEPT من جزء الإجراءات ( أنظر الفصل السادس ) . تختلف التفاصيل الدقيقة لهذه الآلية من جهاز كمبيوتر، ونظام تشغيل معين لجهاز ونظام تشغيل آخر ، الا أنه فى كوبل IBM OS/VS .. يدخل أمر ACCEPT سجلا واحدا من النهاية الطرفية لشغل الكمبيوتر operator's console او من الملف SYSIN . ويخرج امر DISPLAY سجلا واحدا إلى النهاية الطرفية لشغل الكمبيوتر، أو الى الملف SYSOUT .

مثـال ٤ – ٧ :

```
SPECIAL-NAMES.
    C01 IS TOP-OF-INVOICE
    CONSOLE IS OPERATOR-MESSAGE-DEVICE.
```

الاسم الوظيفى CONSOLE هو كلمة من كلمات IBM المحجوزة ، OPERATOR- MESSAGE- DEVICE هو الاسم الذى تشير بقية برنامج مصدر كوبل إلى النهاية الطرفية الخاصة بمشغل الكمبيوتر فى عبارات DISPLAY, ACCEPT:

```
DISPLAY . . . UPON OPERATOR-MESSAGE-DEVICE
ACCEPT . . . FROM OPERATOR-MESSAGE-DEVICE
```

مثـال ٤ – ٨ :

```
SPECIAL-NAMES.
    C01 IS HEADING-AREA
    C02 IS TOTAL-AREA
    SYSIN IS FUNCTION-SELECT-INPUT
    BACK-ORDER-CODING-SEQUENCE IS STANDARD-1
```

لقد اختير FUNCTION- SELECT- INPUT كاسم كوبل لملف المدخلات الخاص SYSIN. ( كلمة من كلمات IBM المحجوزة ). يستخدم FUNCTION- SELECT- INPUT فى عبارة ACCEPT من جزء الإجراءات .

```
ACCEPT . . . FROM FUNCTION-SELECT-INPUT
```

مثـال ٤ – ٩ :

```
SPECIAL-NAMES.
    SYSOUT IS ERROR-LOG.
```

تتجه المخرجات من عبارات DISPLAY ، من جزء الإجراءات إلى ملف مخرجات خاص بالنظام اسمه SYSOUT ( كلمة من كلمات IBM المحجوزة ) . اسم الكوبل لـ SYSOUT يصبح ERROR- LOG أى :

```
DISPLAY . . . UPON ERROR-LOG
```

## ٤ - ٥ قسم المدخلات والمخرجات

يبنى كل تشغيل بيانات الكوبل على مفهوم معالجة السجلات records فى ملفات files . يتم التفكير فى معلومات المدخلات (بغض النظر عن وحدة المدخلات المستخدمة) على أنها تتكون من سجلات مدخلات input records من ملف مدخلات input file. ويتم التفكير فى معلومات المخرجات ( بغض النظر عن وحدة المخرجات المستخدمة على أنها سجلات مخرجات out-put records وثابة تذهب إلى ملف مخرجات output file . ونظراً لمقدرة التخزين المساعد - مثل القرص على الاتصال بأى سجل فى حدود المللى ثانية ( القسم الرابع من الفصل الأول ) - فيمكن لهذا النوع من التخزين ( يسمى بصفة عامة تخزين اتصال مباشر ) أن يدعم ملفات مدخلات ومخرجات فى نفس الوقت بواسطة نفس برنامج الكوبل (input- output file 5) . ويمكن لسجل الاتصال المباشر أن يكون مدخلات ويجرى عليه تعديلاً ثم يصبح مخرجات فى نفس موقعه الأصلى على القرص ، وتسمى مثل هذه العملية التجديد فى نفس الموقع update in place .

**مثـــال ٤ - ١٠ :**

يمكن أن يقوم برنامج كوبل يطبع شيكات رواتب العاملين بتشغيل ٣ ملفات على الاقل .

( ١ ) ملف مدخلات تحتوى سجلاته ( بطاقات الوقت ) على حقول رقم تعريف العامل ، وعدد ساعات العمل الاسبوعية .

( ٢ ) معلومات عن الراتب كمخرجات عن طريق الطابع، تطبع على صيغة للشيكات سابقة الإعداد . يعتبر كل سطر مطبوع سجلا فى ملف مخرجات مطبوع .

( ٣ ) معلومات عن معدل الأجر لكل من العاملين يمكن أن تكون مدخلات من ملف رئيسى للعاملين على قرص . عند إدخال بطاقة الوقت لكل عامل .. يستدعى كذلك سجله من الملف الرئيسى للعاملين كمدخلات من القرص .

وربما يراد تعديل بعض حقول الملف الرئيسى للعاملين لبعض ( أو كل ) العاملين ؛ نتيجة لتشغيل الرواتب مثل إضافة إجمالى الراتب للأسبوع الحالى إلى إجمالى الراتب المدفوع منذ بداية العام حتى تاريخه . يمكن أن يكون السجل الرئيسى للعامل المجدد updated مخرجات ، تسجل فى نفس موقعه الاصلى على القرص ( مع حذف محتوياته السابقة ، التى اصبحت متقادمة الآن ). وعلى هذا يمكن استخدام ملف القرص كملف مدخلات ومخرجات لبرنامج الرواتب ، وهذا موقف معتاد للملفات الموجودة على وحدات تخزين مساعد .

وتكوينيا ... يحتوى قسم المدخلات والمخرجات على مقطع كوبل واحد، وهو مقطع التحكم فى الملفات FILE- CON-TROL، الذى يجب أن يحتوى على عبارة SELECT لكل ملف ( مدخلات أو مخرجات أو مدخلات ومخرجات ) خاص بالبرنامج . كل عبارة SELECT تتكون من أجزاء عديدة، وتنتهى بنقطة ( انظر الاسطر من ١٦ - ٢٢ فى شكل ٤ - ١ ).

## اختــر اســم ملــف

( السطر رقم 16 )

اسم الملف هو اسم يضعه المبرمج ، ويجب ان يكون وصفيا بقدر الامكان، مثل : EMPLOYEE- MASTER- FILE بدلا من FILE-1 . ويستخدم للإشارة إلى الملف فى كل عبارات البرنامج الأخرى .

مثـال ٤ – ١١ :

```
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT CUSTOMER-SALES-FILE...
    SELECT SALES-REPORT...
```

يقوم هذا البرنامج بتشغيل ملفين أثنين بالضبط ( حيث إنه لايوجد سوى عبارتى SELECT فقط ) .

يشار إلى الملفين فى بقية برنامج المصدر بانهما : SALES- REPORT, CUSTOMER- SALES- FILE.

## محـدد لـ اسـم خارجـى

( السطر 17)

يحدد جزء ASSIGN TO الملفات لوحدات المدخلات والمخرجات . ويعطى لكل ملف اسم خارجى، يصاحب ملف الكوبل فى عبارات لغة تحكم العمل، أو فى أوامر نظام التشغيل التى تصف الملف ووحدة المدخلات أو المخرجات أكثر لنظام تشغيل الكمبيوتر ( أنظر القسم رقم ٩ من الفصل الأول ، والمشكلة ٤٥ من الفصل الرابع ) . يتغير شكل ومحتويات عبارات وأوامر تحكم العمل بشدة من جهاز كمبيوتر ونظام تشغيل لجهاز كمبيوتر، ونظام تشغيل آخرين وعليك أن تعرف فى هذه اللحظة كيفية كتابة الأسماء الخارجية للكمبيوتر المتاح لك استخدامه . فى كوبل IBM OS/VS يجب أن يتكون الاسم الخارجى من ١ الى ٨ حروف أبجدية أو أرقام على ان يكون أول رمز حرفا أبجديا . ولمصلحة مبرمج الكوبل أن يتعلم مايستطيع تعلمه عن لغة تحكم العمل .

مثـال ٤ – ١٢ :

```
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT CUSTOMER-SALES-FILE
        ASSIGN TO CUSTSALE...
    SELECT SALES-REPORT
        ASSIGN TO SALERPT...
```

هنا .. يتحدد الاسم الخارجى CUSTSALE ، للملف للملف CUSTOMER - SALES- FILE والاسم الخارجى SALERPT للملف SALES- REPORT . تظهر هذه الاسماء الخارجية فى عبارات تحكم العمل، أو فى أوامر نظام التشغيل، والتى تقدم معلومات أكثر عن وحدات المدخلات والمخرجات اللازمة لتشغيل هذه الملفات .

## التنظيم تتابعى

( السطر رقم 18 : اختيارى )

## حالة الاتصال تتابعية

(السطر رقم 19 : اختيارى)

لغة الكوبل مبنية حول مفهوم تشغيل سجلات الملف سجلا سجلا . وهناك مرونة على أية حال، بغض النظر عن أى السجلات يتم تشغيله أولا وأيها ثانيا .. وهكذا . وهناك حالتان أساسيتان لتشغيل الملف أو الاتصال به وهما:

* **اتصال تتابعى** sequential access ، أو تشغيل تتابعى access processing حيث يتم تشغيل السجلات طبقا لترتيبها الواقعى على وحدة المدخلات او المخرجات، دون حاجة إلى معلومات فى الملف غير سجلات البيانات الواقعية نفسها . ويمكن أداء هذا النوع من التشغيل لملفات موجودة على أى نوع من أنواع وحدات المدخلات والمخرجات .

* **اتصال عشوائى** random access أو تشغيل عشوئى random processing. يتم تشغيل السجلات طبقا للترتيب الذى يحدده البرنامج ، والذى يمكن أن يختلف عن الترتيب الذى سبق أن خزنت به السجلات واقعيا على وحدة المدخلات والمخرجات. وهذا النوع من التشغيل ممكن بالنسبة للملفات الموجودة على وحدات اتصال مباشر فقط ، وقد يتطلب أن يحتوى الملف على معلومات إضافية ، للمساعدة فى تحديد سجلات البيانات التى يطلبها البرنامج .

ويشير تنظيم الملف file organization إلى الطريقة المستخدمة فى تخزين السجلات ( أى معلومات إضافية تلزم للمساعدة فى تحديد موقع السجلات الفردية ) فى الملف . ويقدم الكوبل النمطى للمبرمج ثلاثة خيارات لتنظيم الملف .

* **تنظيم تتابعى** ، sequential organization ، حيث يمكن وضع الملفات المرتبة تتابعيا على أى نوع من أنواع وحدات المدخلات والمخرجات . يحتوى الملف على سجلات بيانات فعلية فقط ، والتى يجب ان يتم تشغيلها بنفس الترتيب الذى وضعت فيه وحدة المدخلات والمخرجات .

* **تنظيم مفهرس** indexed organization : يجب أن توضع الملفات المفهرسة على وحدات اتصال مباشر ويمكن تشغيلها تتابعيا أو عشوائيا . عندما يتم تشغيلها تتابعيا .. تسترجع السجلات فى ترتيب تصاعدى ، طبقا لمفتاح key يسبق تعريفه (وهو حقل تعرف محتوياته كل سجل فى الملف تعريفا فريدا مثل رقم الجزء أو رقم الحساب) . وعندما يتم تشغيلها عشوائيا .. تعرف السجلات باستخدام نفس المفتاح . ولايحتوى الملف على سجلات البيانات الواقعية فقط لكنه يحتوى كذلك على سجلات فهرس index records التى تحتوى على معلومات خاصة بموقع كل سجل بيانات فى الملف . فإذا كان الحقل الرئيسى ( حقل المفتاح) لسجل البيانات معروفا ، فمن الممكن تحديد موقعه بسرعة مؤكدة بالبحث خلال سجلات الفهرس فى الملف ، وهذا مايجعل التشغيل العشوائى ممكنا .

* **تنظيم نسبى** relative organization . لاتعد الملفات النسبية شائعة الاستخدام مثل الملفات التتابعية أو المفهرسة . يحتوى الملف على سجلات بيانات فعلية فقط ، ترقم بالارقام ١ و ٢ و ٣ و... ويرقم السجل record number .. هذا يمكن تشغيل الملف عشوائيا .

وحيث ان مجالات الأعمال معتادة على تعريف المعلومات المحفوظة فى سجلات بواسطة مفتاح ket ، وليس رقم سجل record number.. فلاتستخدم الملفات النسبية إلا عندما يكون الاتصال السريع جدا ضروريا ؛ وإذا يجب أن توضع الملفات النسبية على وحدات اتصال مباشر .

توفر عبارة SELECT وسيلة لمبرمج الكوبل لاختيار تنظيم الملف وحالة الاتصال لكل ملف يراد تشغيله بواسطة البرنامج. هذا الخيار يكون حاسما . عادة ماتنتج الملفات التى تتطلب تشغيلا عشوائيا بتنظيم مفهرس ، أما الملفات التى تتطلب تشغيلا تتابعيا فقط .. فتنتج بتنظيم تتابعى ، وتنتج الملفات التى تتطلب كلاً من التشغيل التتابعى والتشغيل العشوائى بتنظيم مفهرس ( الذى يوفر كلاً من نوعى التشغيل ) . إن تشغيل الملفات تتابعيا مشروح بالتفصيل فى الفصل الحادى عشر ، اما الملفات المفهرسة والنسبية فتقع خارج نطاق هذا الملخص .

**مثـــال ٤ – ١٣ :**

```
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT CUSTOMER-SALES-FILE
        ASSIGN TO CUSTSALE
        ORGANIZATION IS SEQUENTIAL
        ACCESS IS SEQUENTIAL . . .
    SELECT SALES-REPORT
        ASSIGN TO SALERPT
        ORGANIZATION IS SEQUENTIAL
        ACCESS IS SEQUENTIAL . . .
```

هناك عبارتا SELECT للفين :

CUSTOMER- SALES- FILE ( والاسم الخارجى CUSTSALE )

و SALES- REPORT ( والاسم الخارجى SALERPT ).

• ORGANIZATION IS SEQUENTIAL

يمكن أن يحذف هذا الجزء نظرا لأن التنظيم التتابعى للملفات هو التقليدى ، وهناك إمكانيات أخرى، وهى :

- ORGANIZATION IS RELATIVE, ORGANIZATION IS INDEXED .

• ACCESS MODE IS SEQUENTIAL

يمكن حذف هذا الجزء كذلك لان الاتصال التتابعى هو التقليدى ، وهناك إمكانيات أخرى وهى :

ACCESS MODE IS DYNAMICو ACCESS MODE IS RANDOM

الملفات المرتبة تتابعيا يمكن الاتصال بها تتابعيا فقط ، وتتطلب حالات الاتصال الأخرى تنظيما مفهرسا أو تنظيما نسبيا .

## حالـــة الملـــف هـــى

(السطر رقم 20 : اختيارى)

من ناحية المفهوم.. فمعالجة جزء الإجراءات لسجلات الملف فى الكوبل تكون بسيطة جدا ؛ إذ تبدأ عبارة READ بسجل واحد ، بإدخاله من ملف مدخلات من وحدة مدخلات أو وحدة تخزين مساعد فى ذاكرة وحدة التشغيل المركزية للتشغيل ، وتنقل عبارة WRITE سجلا من ذاكرة وحدة التشغيل المركزية الى وحدة مخرجات او وحدة تخزين مساعد، يكون بها ملف المخرجات . وقبل ان يمكن استخدام عبارة READ ، أو عبارة WRITE فى تشغيل السجلات الموجودة فى ملف .. يجب أن تستخدم عبارة OPEN لاعداد الملف للتشغيل ( انظر مثال ٢ – ٥ ) . وبالمثل يجب أستخدام عبارة CLOSE لإنهاء نشاط الملف بعد الانتهاء من تنفيذ كل عبارات WRITE, READ .

وفى الحياة الواقعية .. فإنه من السهل على أية حال حدوث أخطاء للاشياء . فمثلا قد تفشل عملية OPEN نظرا لعدم توافق وصف الملف فى برنامج مصدر الكوبل، مع الوصف المصاحب فى عبارات تحكم العمل المصاحبة . ويمكن ان تفشل عملية WRITE فى اضافة سجل إلى ملف ، مرتب ترتيبًا تتابعيًا على قرص، وذلك لسبب عدم وجود مكان متاح على القرص. وفى الواقع .. فإن توقع فشل معين يكون مبنيا داخل برامج الكوبل. وعلى هذا.. لتشغيل السجلات فى ملف تتابعى.. فإننا نصمم على ذلك البرنامج الذى يحفظ قراءة سجل، ويشغل السجل، ويخرج النتائج حتى تفشل READ بسبب الانتهاء من قراءة كل سجلات الملف ( وهذا يسمى شرط نهاية الملف ).

يجب أن تكتب البرامج لمعالجة الشروط الاستثنائية exception conditions التى يمكن ان تحدث أثناء التشغيل . وتوفر عبارة SELECT وسيلة للحصول على تغذية مرتجعة feedback ، خاصة بنجاح او فشل كل عملية يتم إجراؤها على الملف . وهذه التغذية المرتجعة تقدم على هيئة منطقة بيانات ينقل فيها شفرة معينة بعد كل عملية من عمليات WRITE, READ, CLOSE, OPEN تجرى على الملف . وتذكر الشفرة ما إذا كانت العبارة ناجحة أم لا – فإذا لم تكن ناجحة فما هو السبب .

**مثـــال ٤ – ١٤ :**

```
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT MONTHLY-SALES-FILE
        ASSIGN TO MSALES
        FILE STATUS IS MONTHLY-SALES-FILE-STATUS
```

فى جزء FILE STATUS أختار المبرمج اسم MONTHLY- SALES- FILE- STATUS لمنطقة البيانات التى تستخدم فى توفير التغذية المرتجعة الخاصة بنتائج كل عملية من عمليات جزء الإجراءات (OPEN, WRITE, READ, CLOSE ) التى تم تنفيذها على الملف . يجب أن يعرف MONTHLY- SALES- FILE- STATUS فى جزء البيانات كمنطقة يمكن أن تحتفظ بشفرة من خانتين تسجل نجاح أو فشل أى عملية معطاه .. على الملف . أفحص دليل المورد لمعنى FILE STATUS ، الخاص بالنظام المتاح لك استخدامه .

لايكلف الكثير من مبرمجى الكوبل ذوى الخبرة أنفسهم تعريف منطقة FILE STATUS بالنسبة للملفات المرتبة تتابعيا ، حيث إن هناك طرقاً أخرى للتأكد من اتمام عمليات المدخلات والمخرجات بنجاح . ( انظر الفصل الحادى عشر ).

## حجــز مناطــق

(السطر رقم 21 : اختيارى)

يمكن استخدام هذا الجزء للإسراع من تشغيل الملف عندما يكون تتابعيا ( انظر الذاكرات المتعددة – ٥ – ٨)

## ٤ – ٦ أمثلة على جزء الأوساط

### مثـال ٤ – ١٥

```
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-370.
OBJECT-COMPUTER. IBM-370.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT OLD-INVENTORY-MASTER-FILE
        ASSIGN TO OLDMAST.
    SELECT NEW-INVENTORY-MASTER-FILE
        ASSIGN TO NEWMAST.
    SELECT TRANSACTION-FILE
        ASSIGN TO TRANS.
```

هذا جزء أوساط لبرنامج تجديد ملف تتابعى يستخدم سجلات ملف العمليات الجارية TRANSACTION FILE فى تجديد المعلومات ، فى سجلات ملف المخزون الرئيسى القديم OLD- INVENTORY- MASTER- FILE . وتوضع السجلات التى تحتوى على التجديد فى ملف المخزون الرئيسى الجديد NEW- INVENTORY- MASTER- FILE .

كل الملفات الثلاثة تتابعية (الشىء التقليدى عندما يكون جزء ORGANIZATION IS محذوفا) ، ويتم تشغيلها تتابعيا (الشىء التقليدى عندما يكون جزء ACCESS MODE IS محذوفاً) .

وحيث إن كل الملفات تتابعية ، فإن مناطق شفرة FILE STATUS تكون اختيارية ولاتستخدم ( يجرى اختبار للشروط الإستثنائية أثناء عمليات المدخلات والمخرجات باستخدام طرق جزء الإجراءات الأخرى ) .

تسلسل التتابع يكون تقليديا لنظام الكمبيوترEBCDIC لنظام IBM 370 . وحيث انه لم يتحدد حالة تصحيح DE-BUGGING MODE ، تعامل الاسطر الموجود حرف D فى عمودها السابع كتعليقات ولايكون لها أى تاثير على تنفيذ البرنامج . لأسماء ASSIGN TO الشكل الصحيح بالنسبة لنظم IBM OS/VS (TRANS, NEWMAST, OLDMAST) تحتوى على رموز تتراوح من ١ – ٨ وهى حروف أو أرقام فقط وأول رمز حرف) .

مثـال ٤ - ١٦ :

```
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-370
    WITH DEBUGGING MODE.
OBJECT-COMPUTER. IBM-370.
SPECIAL-NAMES.
    C01 IS TOP-OF-CHECK
    C02 IS PAYEE-AREA
    C05 IS AMOUNT-AREA

INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT PAYROLL-FILE
        ASSIGN TO PAYROLL
        ORGANIZATION IS SEQUENTIAL
        ACCESS MODE IS SEQUENTIAL
        FILE STATUS IS PAYROLL-STATUS-CODE

    SELECT PAYCHECK-FILE
        ASSIGN TO CHECKS
        ORGANIZATION IS SEQUENTIAL
        ACCESS MODE IS SEQUENTIAL
        FILE STATUS IS CHECK-STATUS-CODE
```

## تعليقـــــات :

( ١ ) محدد حالة تصحيح DEBUGGING MODE ، وعلى هذا تترجم الاسطر الموجود حرف D فى عمودها السابع ويكون لها تأثير أثناء التنفيذ.

( ٢ ) لاحظ استخدام النقاط المرتبة فى إنهاء الاسماء الخاصة وعبارة الاختيار SELECT.

( ٣ ) القنوات رقم ١ ، ٢ ، ٣ لآلية التحكم فى عربة الطباعة، تستخدم فى وضع الطابع ( نقل الطابع ) إلى الثلاثة مناطق من الشيكات المراد طباعتها بواسطة البرنامج .

( ٤ ) يستخدم البرنامج ملفين . باستخدام معلومات المدخلات من PAYROLL- FILE تتم طباعة الشيكات فى ملف PAYCHECK- FILE . للأسماء الخارجية للملفات PAYROLL & CHEKS) نفس التشكيل المطلوب من نظم IBM OS/VS

(0) جزئى ORGANIZATION & ACCESS MODE محددان، بالرغم من إن لهما القيمة التقليدية SEQUEN- TIAL. وقد تم عمل ذلك لتقديم توثيق واضح للبرنامج .

( ٦ ) صاحب كل من الملفين مناطق شفرة FILE STATUS ، التى يمكن اختبارها بعد كل عملية على ملف فى جزء الإجراءات. يجب أن يزداد تعريف كل من PAYRPLL- STATUS- CODE, CHECK- STATUS- CODE فى جزء البيانات .
وكان من الممكن حذف مناطق FILE STATUS ومعالجة شروط الاستثناء بطرق أخرى ( انظر الفصل الحادى عشر).

## اسـئلة مراجعـة

٤ -١ صف تكوين جزء الأوساط .

٤ -٢ ما الغرض من قسم التشكيل ؟

٤ -٣ ما الغرض من قسم المدخلات والمخرجات ؟

٤ - ٤ اذكر قواعد كتابة المدخلات والمخرجات .

٤ - ٥ وضح الغرض من اسطر التصحيح ، وكيف تتأثر بمقطع كمبيوتر المصدر ؟

٤ - ٦ ما معنى تسلسل التتابع ؟

٤ - ٧ كيف يفسر تسلسل التتابع التقليدى ؟

٤ - ٨ ماذا يجب عمله ليمكن استخدام تسلسل تتابع غير تقليدى ؟

٤ - ٩ ما الوظائف الثلاث التى يؤديها مقطع الاسماء الخاصة ؟

٤ - ١٠ ما المقصود بالصيغ المستمرة ؟

٤ - ١١ ما المقصود بشريط تحكم عربة الطباعة ؟

٤ - ١٢ وضح كيف تربط قنوات تحكم العربة بمناطق الصيغ المستمرة .

٤ - ١٣ ماذا تستخدم بعض الطابعات الحديثة بدلا من شريط تحكم عربة الطباعة ؟

٤ - ١٤ ما تسلسل التتابع المعروف بانه STANDARD-1 وما تسلسل التتابع المعروف بانه NATIVE فى نظم IBM ؟

٤ - ١٥ ما النهاية الطرفية لمشغل الكمبيوتر الآدمى ؟

٤ - ١٦ ماذا تفعل عبارات DISPLAY, ACCEPT من جزء الإجراءات ؟ كيف ترتبط هذه العبارات بمقطع الأسماء الخاصة من جزء الأوساط ؟

٤ - ١٧ اذكر معنى الأسماء الوظيفية التالية فى كوبل IBM OS/VS

C04 ( هـ ) C01 ( د ) SYSOUT ( جـ ) CONSOLE ( ب ) SYSIN .

٤ - ١٨ ما هى وحدات الاتصال المباشر

٤ - ١٩ ماهى عملية التجديد فى نفس الموقع ؟

٤ - ٢٠ اذكر بعض الامثلة للسجلات والملفات .

٤ - ٢١ صف كل جزء من اجزاء عبارة SELECT واشرح مايفعله .

٤ - ٢٢ ما هى لغة تحكم العمل JCL ؟

٤ - ٢٣ ما هى لغة اوامر نظام التشغيل ؟

٤ - ٢٤ كيف يرتبط جزء ASSIGN TO من عبارة SELECT بلغة تحكم العمل ؟

٤ - ٢٥ وضح اتصال الملف تتابعيا ( أو تشغيل الملف تتابعيا ) ؟.

٤ - ٢٦ وضح اتصال الملف عشوائيا ( أو تشغيل الملف عشوائيا ) .

٤ - ٢٧ عرف تنظيم الملف ؟

٤ - ٢٨ وضح بإيجاز ترتيب ( تنظيم ) الملف تتابعيا .

٤ - ٢٩ وضح بإيجاز ترتيب ( تنظيم ) الملف المفهرس .

٤ - ٣٠ وضح بإيجاز ترتيب ( تنظيم ) الملف النسبى .

٤ – ٣١ لماذا يعتبر اختيار تنظيم الملف مهما ؟

٤ – ٣٢ اربط الحاجة الى اتصال تتابعى والحاجة الى اتصال عشوائى باختيار تنظيم الملف ؟

٤ – ٣٣ وضح بإيجاز الغرض من عبارات جزء الإجراءات التالية :

CLOSE, WRITE, READ, OPEN

٤ – ٣٤ اذكر بعض شروط الاستثناءات التى يمكن أن تحدث أثناء تشغيل الملف .

٤ – ٣٥ كيف يشير جزء FILE STATUS من عبارات SELECT الى معالجة شروط استثناءات ؟

٤ – ٣٦ ما هو شرط نهاية الملف ؟

# مسائل محلولة

٤ – ٣٧ أكتب جزء اوساط لبرنامج يختبر على نظام كمبيوتر IBM 370 . اسطر التصحيح سوف تترجم الى برنامج الهدف ويستخدم تسلسل التتابع EBCDIC .

```
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-370 WITH DEBUGGING MODE.
OBJECT-COMPUTER. IBM-370.
```

أو

```
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-370
                 WITH DEBUGGING MODE.
OBJECT-COMPUTER. IBM-370
                 PROGRAM COLLATING SEQUENCE IS EBCDIC-ORDER.
SPECIAL-NAMES.
                 EBCDIC-ORDER IS NATIVE.
```

٤ – ٣٨ اذكر كل الأشكال المختلفة الصحيحة للسطر الخامس، فى شكل (٤ – ١)

(1) PROGRAM COLLATING SEQUENCE IS alphabet-name; (2) PROGRAM SEQUENCE IS alphabet-name; (3) PROGRAM SEQUENCE alphabet-name; (4) COLLATING SEQUENCE IS alphabet-name; (5) SEQUENCE IS alphabet-name; (6) PROGRAM COLLATING SEQUENCE alphabet-name; (7) COLLATING SEQUENCE alphabet-name; (8) SEQUENCE alphabet-name; (9) blank line.

٤ – ٣٩ يراد طباعة شيكات بواسطة الكمبيوتر . وقد اعدت قنوات تحكم العربة لتتفق مع مناطق موجودة على الشيك على النحو التالى : القناة ١ للتاريخ date القناة ٢ للمستفيد من الشيك payee والقناة ٥ للكمية amount . ويعرض البرنامج على نهاية مشغل الكمبيوتر الطرفية إجمالى عدد الشيكات المطبوعة كذلك . اكتب مقطع الأسماء الخاصة لمثل هذا البرنامج.

```
SPECIAL-NAMES.
    C01 IS CHECK-DATE-AREA
    C02 IS CHECK-PAYEE-AREA
    C05 IS CHECK-AMOUNT-AREA
    CONSOLE IS CHECK-CONTROL-COUNT-DEVICE
```

٤ - ٤٠ اكتب قسم التشكيل لبرنامج مستخدما تسلسل التتابع التالى :

B بعدها J ثم K, A ..., W, C ... I , 7, 1, 0, 2 ... 6, 9 .

تم تصحيح ، البرنامج تماما ويراد ترجمته وتنفيذه على كمبيوتر من طراز IBM 370

```
CONFIGURATION SECTION.

SOURCE-COMPUTER. IBM-370.

OBJECT-COMPUTER.  IBM-370
                  COLLATING SEQUENCE IS
                      WAREHOUSE-BIN-ORDER.

SPECIAL-NAMES.

    WAREHOUSE-BIN-ORDER IS
        "B" "J" "A"
        "K" THRU "W"
        "C" THRU "I"
        "7" "1" "0"
        "2" THRU "6"
        "9"
```

٤ - ٤١ اكتب قسم التشكيل لبرنامج ينفذ على كمبيوتر من طراز IBM 370 ، وله تسلسل تتابع ASCII، بدلا من تسلسل التتابع التقليدى EBCDIC ، ويجب أن تترجم أسطر التصحيح .

```
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-370 DEBUGGING MODE.
OBJECT-COMPUTER.  IBM-370 SEQUENCE ASCII-ORDER.
SPECIAL-NAMES.
    ASCII-ORDER IS STANDARD-1.
```

٤ - ٤٢ عادة ماتسمى قارئات البطاقات ، والطابعات، ومثقبات البطاقات بوحدات وحدة السجل unit record devices .

اشرح مفاهيم الملف والسجل والحقل بالنسبة الى ( أ ) مجموعة بطاقات . ( ب ) تقرير تحليل مبيعات مطبوع على صيغ مستمرة .

(أ) كل بطاقة عبارة عن سجل، ومجموعة البطاقات هى الملف ، وكل قطعة معلومات منفصلة مثقبة على البطاقة هى حقل، وذلك مثل رقم البطاقة أو الكمية المطلوبة .

(ب) مجموعة كل الاسطر المطبوعة فى التقرير هى الملف ، وكل سطر عبارة عن سجل ، وكل قطعة معلومات على السطر عبارة عن حقل ( بما فى ذلك الفراغات الفاصلة ) .

٤ - ٤٣ صف كيف يمكن عمل تجديد سجل قرص فى موقعه ، بالنسبة إلى عمليات حسابات توفير البنك .

اعتبر أن هناك بنكًا له نهايات طرفية للصرف فى وسط الخط المفتوح ( أى أن النهايات الطرفية متصلة بنظام كمبيوتر البنك، أو بواسطة كابل كهربائى أو بواسطة اتصالات برقية ) . إذا أراد أحد العملاء عمل إيداع أو سحب من حساب التوفير، فيكتب الصراف مستخدما النهاية الطرفية رقم الحساب ونوع العملية الجارية والكمية المتعامل بها .

وهذه المعلومات هى مدخلات بواسطة برنامج كمبيوتر، يتصل عشوائيا بسجل ملف قرص به موازنة حساب التوفير لهذا العميل (للسماح بالتشغيل العشوائى يمكن أن يكون الملف مفهرسا طبقا لارقام حسابات التوفير ) . يجدد عند ذلك البرنامج موازنة الحساب بإضافة أو طرح كمية العملية الجارية . وأخير يكتب سجل العميل المجدد فى موقعه الاصلى على القرص .

٤ – ٤٤ حدد أسماء ملفات الكوبل التالية ، وما إذا كانت صحيحة أم صحيحة لكن ضعيفة، أم غير صحيحة .

(*a*) SELECT REPORT ASSIGN TO...

(*b*) SELECT REPORT-FILE ASSIGN TO...

(*c*) SELECT OUT-OF-STOCK-ITEMS-REPORT ASSIGN TO...

(*d*) SELECT INPUT-FILE ASSIGN TO...

( أ ) غير صحيح : كلمة REPORT هى كلمة محجوزة .

( ب ) صحيح لكنه ضعيف : REPORT- FILE ليس وصفيا .

( جـ) صحيح .

( ى ) صحيح لكنه ضعيف : ربما يكون هناك عديد من ملفات المدخلات .

٤ – ٤٥ يجب ان يوصف كل ملف يستخدم فى برنامج كوبل بعبارة SELECT فى قسم المدخلات والمخرجات . ما الغرض من ASSIGN TO فى عبارة SELECT ؟

جزء الأوساط ( حيث أنه يتعامل مع أوساط نظم مكونات البرنامج ) هو الجزء الأكثر إحتمالا للتغيير من برنامج الكوبل إذا غيرت الكمبيوتر المستخدم .

**ونعطى هنا مثالين :**

( أ ) يتطلب كوبل IBM OS/VS ( لنظم IBM 370 ) .. إنه يجب أن يوصف كل ملف وصفا إضافيا فى عبارة تحكم عمل، تسمى تعريف بيانات DD (data- definition) . واسم ASSIGN TO هو الواصل المشترك بين عبارة DD والمعلومات فى برنامج مصدر الكوبل . وعلى هذا فللعبارة :

SELECT INVENTORY- MASTER ASSIGN TO INVMAST..

يمكن أن تكون عبارة DD المصاحبة على النحو التالى :

| عبارة DD | توضيحها |
|---|---|
| INVMAST DD UNIT= DISK, | // الوحدة الفعلية هى قرص |
| VOL= SER= ABC123, | // مجموعة الأقراص اسمها ABC123 |
| DISP= OLD, | // الملف موجود فعلا |
| DSNAME= INVNTRY. MASTER. A17 | // اسم الملف فى عنوان الملف |

وجود ASSIGN TO وهو INVMAST فى بداية عبارة DD يكون إجباريا .

( ب) يستخدم كوبل Burroughs ( لنظم كمبيوتر B1910 ) لغة أوامر نظام تشغيل بدلا من لغة تحكم العمل . ويستطيع المستفيد الموجود أمام نهاية طرفية أن يكتب برنامج مصدر كوبل، ثم يكتب عبارة لغة أوامر تتسبب فى ترجمة البرنامج .

افرض أن برنامج المصدر به عبارة SELECT التالية :

SELECT INVENTORY ASSIGN TO DISK

اسماء ملفات B1910 محددة بعدد عشرة رموز (INVENTORY) . DISK كاسم فى ASSIGN TO وهى كلمة محجوزة تحدد للنظام أن INVENTORY موجود على وحدة قرص مغناطيسى . وفى كوبل B1910 .. يخدم اسم الملف -IN VENTORY كواصل بين تعريف الملف فى البرنامج، وعبارات لغة الأوامر والتى تعرفاًاكثر . فاذا كان تعريف البرنامج هو INVENTORY فيمكن أن يكتب الأمر التالى لتنفيذ البرنامج المترجم :

EXECUTE INVUPDATE, FILE INVENTORY NAME INVMASA17

يطلب امر EXECUTE من نظام التشغيل ان ينفذ برنامج الهدف المحدد . ويجعل جزء FILE الملف المسمى -INVENTO RY على القرص . وعندما ينفذ البرنامج .. يتجدد ملف القرص INVMASA17..

٤ – ٤٦ حدد اختياراتك لتنظيم الملفات، وحالات الاتصال بالنسبة للتطبيقات التالية :

( أ ) ملف رئيسى لحد المديونية لشركة تسمح للتجار الآخرين بالاتصال هاتفيا لاسترجاع حد مديونية بطاقات ائتمان العملاء .

( ب ) ملف شريط لاسماء وعناوين تستخدم فى طباعة العناوين البريدية .

( جـ) ملف قرص لحسابات المدينين به سجلات بموازنات العملاء الحالية .

( أ ) التنظيم : مفهرس ( طبقا لرقم بطاقة الائتمان ) الاتصال عشوائي .

( ب) التنظيم : تتابعى ( الخيار الوحيد لوحدات الاتصال غير المباشر ) . الاتصال : تتابعى ( الاختيار الوحيد للشريط).

( جـ) التنظيم : إما مفهرس ( طبقا لرقم العميل ) أو تتابعى ( مرتب طبقا لرقم العميل ) . الاتصال : يجب أن يجدد الملف بمعلومات عن المدفوعات والمشتروات الجديدة . فإذا كان التنظيم مفهرسا ، فيمكن أن يحدث هذا عشوائيا . أما إذا كان ملف المدفوعات والمشتروات مرتبا كذلك طبقا لرقم العميل ، يمكن أن يتحقق التجديد بالاتصال العشوائى التتابعى . وتتم طباعة التقارير ( مثل الموازنات القديمة ) باستخدام الاتصال التتابعى .

٤ – ٤٧ ما المقصود بشرط نهاية الملف ؟

يطبق مفهوم شرط نهاية الملف عندما يتم تشغيل الملف تتابعيا فى المدخلات . يقرأ أول سجل فى الملف أولا، و يليه السجل الثانى فالثالث... وهكذا . فإذا كان فى الملف 7000 سجل فتحدث نهاية الملف عندما تحدث محاولة القراءة 7001 وتفشل .

٤ – ٤٨ اكتب جزء أوساط كاملا لبرنامج لطباعة شيكات . يستخدم البرنامج ملف شريط للمدخلات به سجلات الوقت الاسبوعية مرتبًا طبقا لرقم العامل ، ويستخدم ملف عاملين رئيسيًا على قرص كمدخلات كذلك ، مرتبًا ، وطبقا لرقم العامل . كل تشغيل الملفات يحدث تتابعيا ، والبرنامج سبق تصحيحه . لاتستخدم قنوات تحكم العربة فى هذا البرنامج ، وذلك بالرغم من أن بعض الرسائل يمكن أن تظهر فى النهاية الطرفية لمشغل الكمبيوتر.

```
ENVIRONMENT DIVISION.

CONFIGURATION SECTION.

SOURCE-COMPUTER. IBM-370.

OBJECT-COMPUTER. IBM-370.

SPECIAL-NAMES.

    CONSOLE IS OPERATOR-MESSAGES-DEVICE


INPUT-OUTPUT SECTION.

FILE-CONTROL.

    SELECT WEEKLY-TIME-FILE
        ASSIGN TO WKLYTIME
        ORGANIZATION IS SEQUENTIAL
        ACCESS IS SEQUENTIAL
        FILE STATUS IS TIME-FILE-STATUS

    SELECT EMPLOYEE-MASTER-FILE
        ASSIGN TO EMPMAST
        ORGANIZATION IS SEQUENTIAL
        ACCESS IS SEQUENTIAL

    SELECT PAYCHECK-FILE
        ASSIGN TO CHECKS
        ORGANIZATION IS SEQUENTIAL
        ACCESS IS SEQUENTIAL
```

( هل نسيت أن الشيكات المطبوعة تكون ملفا، وتتطلب على هذا عبارة SELECT ؟ ملاحظة : ( ١ ) الاسطر الفارغة والنقاط المرتبة . ( ٢ ) حذف DEBUGGING MODE و COLLATING SEQNENCE . ( ٣ ) مصاحبة اسم الكوبل OPERATOR- MESSAGE- DEVICE مع CONSOLE للنظام ( ٤ ) حددت منطقة اسمها TIME- FILE- STATUS للملف WEEKLY- TIME- FILE، وعرفت أكثر فى جزء البيانات.. حيث يوضع فيها شفرة تغذية مرتجعة بعد كل عملية تجرى على الملف .

## تمارين برمجة

من ٤٩ إلى ٥٨ عدل التمارين من ٢٩ – ٣٨ فى الفصل الثانى لتعكس ماتعلمته عن جزء الأوساط فيها .

# الفصل الخامس

# جــــزء البيـــانات

# Data Division

## ٥ - ١ هيكل جزء البيانات

يحدد شكل ٥ . ١ الهيكل العريض لجزء البيانات ، والذى سبق وصف وظيفته العامة فى القسم الأول من الفصل الثانى .

```
DATA DIVISION.
[FILE SECTION.
 [file description entry
  record description entry] . . . ]
[WORKING-STORAGE SECTION.
 [data item or "record" description entry] . . . ]
```

شكل (٥ - ١)

مع أن كلاً من قسم الملفات FILE SECTION ، وقسم مخزن العمل WORKING- STORAGE SECTION اختيارى، إلا إنهما مطلوبان بصفة عامة عمليا ، حيث يستخدم كل برنامج تقريبا ملفاً واحداً على الأقل وبعض عدادات ، وشعارات ، ومناطق عمل فى مخزن العمل . ويوضح مثال ٥ - ١ جزء بيانات فعليًا مناظراً لجزء الأوساط مثال ٤ - ١ .

**مثال ٥ - ١ :**

```
DATA DIVISION.

FILE SECTION.

FD  CUSTOMER-SALES-FILE
    BLOCK CONTAINS 0 RECORDS
    RECORD CONTAINS 80 CHARACTERS
    LABEL RECORDS ARE STANDARD
    .

01  CUSTOMER-SALES-RECORD.
    05  CUST-SALES-CUSTOMER-NO     PIC X(4).
    05  CUST-SALES-SALES-ID        PIC X(6).
    05  CUST-SALES-AMOUNT          PIC S9(7)V99.
    05  FILLER                     PIC X(61).
```

```
       FD  SALES-REPORT
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .

       01  SALES-REPORT-LINE              PIC X(132).

       WORKING-STORAGE SECTION.

       01  END-OF-FILE-AND-STATUS-AREA.

           05  CUSTOMER-SALES-FILE-END    PIC X.
               88  NO-MORE-CUSTOMER-RECORDS                 VALUE "T".
           05  CUST-SALES-FILE-STATUS     PIC XX.
           05  SALES-REPORT-STATUS        PIC XX.


       01  TOTAL-AND-COMPARISON-AREAS.

           05  OLD-CUSTOMER-NO            PIC X(4).
           05  CUSTOMER-TOTAL             PIC S9(9)V99  COMP-3.
           05  GRAND-TOTAL                PIC S9(9)V99  COMP-3.

       01  COUNTER-AREA.

           05  LINE-COUNT                 PIC S9(3)     COMP-3.
           05  PAGE-COUNT                 PIC S9(5)     COMP-3.
           05  LINE-SKIP                  PIC S9(3)     COMP-3.

       01  HEADING-LINE.

           05  FILLER                     PIC X(10)     VALUE SPACES.
           05  FILLER                     PIC X(20)
                                      VALUE "SALES BY CUSTOMER".
           05  FILLER                     PIC X(10)     VALUE SPACES.
           05  OUTPUT-PAGE-COUNT          PIC ZZ,ZZ9.
           05  FILLER                     PIC X(86)     VALUE SPACES.

       01  ERROR-LINE.

           05  ERROR-CUSTOMER-NO          PIC X(4).
           05  FILLER                     PIC X(2)      VALUE SPACES.
           05  FILLER                     PIC X(22)
                                      VALUE "** OUT OF SEQUENCE **".
           05  FILLER                     PIC X(104)    VALUE SPACES.

       01  OUTPUT-TOTAL-LINE.

           05  FILLER                     PIC X(30)     VALUE SPACES.
           05  OUTPUT-CUSTOMER-TOTAL      PIC $$$$,$$$,$$$.99-.
           05  FILLER                     PIC XX        VALUE " *".
           05  GRAND-TOTAL-FLAG           PIC X.
           05  FILLER                     PIC X(83)     VALUE SPACES.

       01  OUTPUT-LINE.

           05  OUTPUT-CUSTOMER-NO         PIC X(4).
           05  FILLER                     PIC X(10)     VALUE SPACES.
           05  OUTPUT-SALES-ID            PIC X(6).
           05  FILLER                     PIC X(10)     VALUE SPACES.
           05  OUTPUT-AMOUNT              PIC $$,$$$,$$$.99-.
           05  FILLER                     PIC X(88)     VALUE SPACES.
```

# ٥ - ٣ قسم الملفات

يجب أن يحتوى قسم الملفات على جزء وصف الملف file descritiptin entry (FD) لكل ملف، له عبارة SELECT فى جزء الأوساط .

**مثال ٥ - ٢ :**

```
line 52:  SELECT CUSTOMER-SALES-FILE
line 59:  SELECT SALES-REPORT
```

يسمى الملفان المستخدمان فى البرنامج ، وتبدأ محتويات FD المناظرة فى مثال ٥ - ١ على الأسطر (٧٠ - ٨٢) .

بعد معلومات FD .. يجب أن يوجد لكل ملف مستوى 01 واحد على الأقل خاص بوصف السجل record description entry يصف تخطيط السجلات فى هذا الملف . والهيكل التكوينى لمحتوى FD فى شكل ٥ - ٢ .

```
FD  file-name
    [BLOCK CONTAINS [integer-1 TO] integer-2 {CHARACTERS}]
                                             {RECORDS   }
    [RECORD CONTAINS [integer-3 TO] integer-4 CHARACTERS]
    LABEL {RECORD IS  } {STANDARD}
          {RECORDS ARE} {OMITTED }
    [DATA {RECORD IS  } data-name-3 [data-name-4] . . . ]
          {RECORDS ARE}
    [LINAGE IS {data-name-5} LINES
               {integer-5  }
        [WITH FOOTING AT {data-name-6}]
                         {integer-6  }
        LINES AT TOP {data-name-7}
                     {integer-7  }
        LINES AT BOTTOM {data-name-8}
                        {integer-8  }  ]
```

شكل (٥ - ٢)

يجب أن يبدأ كتابة عنوان وصف الملف "FD" فى المنطقة A ، أما بقية الأجزاء فتكتب فى المنطقة B . اسم الملف ( وهو نفس الاسم المستخدم فى عبارة SELECT ) ، يجب أن يظهر أولا ، بعد FD مباشرة . ويمكن ظهور بقية الأجزاء بأى ترتيب . يجب استخدام الأسطر الفارغة ( أو أسطر التعليقات الفارغة ) والترحيل لتسهيل القراءة . ولاتوجد الا نقطة واحدة فقط ، موضوعة فى نهاية محتويات FD ( من الأفضل جعلها نقطة مرتبة ) .

**مثال ٥ - ٣ :**

```
DATA DIVISION.

FILE SECTION.

FD  WEEKLY-PERSONNEL-REPORT

        BLOCK CONTAINS 1 RECORD
        RECORD CONTAINS 132 CHARACTERS
        LABEL RECORDS ARE OMITTED
        DATA RECORD IS PERSONNEL-REPORT-RECORD
        LINAGE IS 50 LINES
            WITH FOOTING AT 45
            LINES AT TOP 8
            LINES AT BOTTOM 8

01  PERSONNEL-REPORT-RECORD . . .
```

لاحظ استخدام الأسطر الفارغة والترحيل والنقطة المرتبة . أجزاء وصف السجل التى تتبع FD بوصف كل حقل من حقول سجلات الملف . يجب أن يكون اسم FD وهو WEEKLY- PERSONAL- REPORT نفس الاسم المستخدم فى عبارة SELECT للملف .

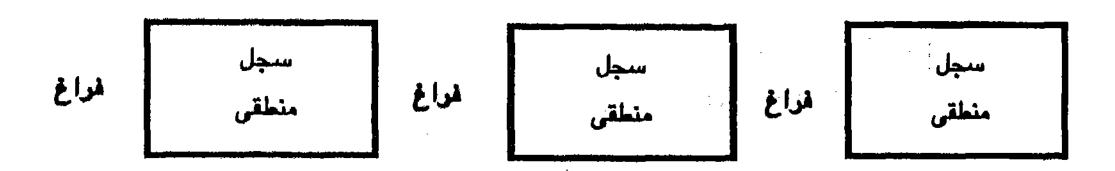
فى الأقسام من ٣ إلى ٧ من هذا الفصل.. نناقش أجزاء FD كما هو مبين فى شكل ٥ - ٢ جزءاً جزءاً . تذكر أن الغرض من كل جزء هو تقديم معلومات خاصة بكيفية تخزين سجلات الملف فى الوحدات الواقعية .

## ٥ - ٣ وصف الملف وما تحتويه المجموعة

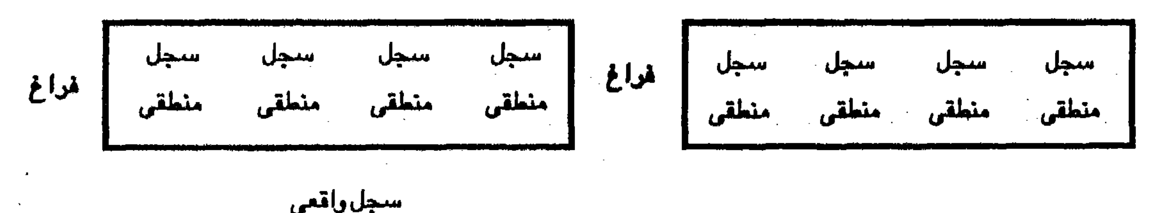
عادة ماتجمع blocked الملفات الموجودة على أقراص، أو شرائط على هيئة مجموعات، لتحقيق أستغلال أكثر كفاءة للمكان والإسراع من تشغيل الملف . وفى عملية التجميع .. ميز بين السجل المنطقى logical record ( الذى يجرى تشغيلا لواحد فقط منه، فى نفس الوقت بواسطة البرنامج )، أو السجل الواقعى physical ( وحدة المعلومات التى تنقل من والى وحدة المدخلات والمخرجات ) .

**مثال ٥ - ٤ :**

الملف الذى لايحتوى على مجموعات الموجود على شريط أو قرص يحتوى على سجلات منطقية ، منفصلة عن بعضها البعض بفراغات مابين السجلات interrecord gaps :




أما فى السجلات المحتوية على مجموعات .. فتجمع السجلات المنطقية على هيئة سجلات واقعية، منفصلة عن بعضها البعض بفراغات مابين المجموعات interblock gaps .




سجل واقعى

فى هذا الملف تجمع فيه السجلات أربعة سجلات منطقية فى كل سجل واقعى .، ونقول على هذا إن له معامل تجميع blocking factor (BF) يساوى ٤ ، وبالنسبة إلى الملف غير المجمع فان معامل التجميع يساوى ١ . ولاى ملف تقرأ أو تكتب كل الرموز الموجودة بين فراغين متتالين كوحدة واحدة . وعلى هذا.. يتطلب الملف المجمع – فى التشغيل التتابعى – $^1/_4$ عمليات المدخلات والمخرجات فقط التى يتطلبها الملف غير المجمع، وينخفض إجمالى وقت وضع القرص أو الشريط إلى حوالى $^1/_4$ الوقت المخصص للملف المجمع . وحيث أن للملف المجمع $^1/_4$ الفراغات فقط .. فالمكان غير المشغول منه يقل عن الملف غير المجمع .

**مثـال ٥ – ٥ :**

افرض إنه مطلوبه عمل تشغيل عشوائى لملف غير تتابعى به ٢٠٠٠٠ سجل مخزون . افرض كذلك أن حوالى نصف السجلات فقط يتم تشغيله فى المتوسط ، فى كل مرة من مرات تنفيذ البرنامج . اذا كان الملف غير مجمع .. فإن ذلك يتطلب حوالى ١٠٠٠٠ عملية مدخلات، للحصول على ١٠٠٠٠ سجل منطقى، يجرى عليها التشغيل .

إذا كان معامل التجميع للملف هو ١٠ ، فمازال الملف فى حاجة إلى حوالى ١٠٠٠٠ عملية مدخلات للحصول على ١٠٠٠٠ سجل منطقى ( لإنه لانتوقع فى التشغيل العشوائى أن يكون السجل التالى المطلوب للتشغيل موجودا فى نفس المجموعة التى سبقت قراءتها ) يمكن أن يكون وقت نقل البيانات للسجلات الإضافية ( غير المشغلة ) فى المجموعات مفقودا . وبالرغم من أن التجميع يمكن أن يسرع من تشغيل الملفات التتابعية .. إلا إنه يبطىء قليلا من التشغيل العشوائى كما يمكن أن يستخدم التجميع ملفات التشغيل العشوائى بغرض توفير أماكن التخزين .

يحدد BLOCK CONTAINS ما إذا كان الملف مجمعا أم لا ، ويحدد معامل التجميع . ويجمع نظام التشغيل تلقائيا السجلات المنطقية مجموعة من المخرجات، ويحلل المجموعة إلى السجلات المنطقية فى المدخلات . وعلى هذا .. يكتب جزء الاجراءات دائما للعمل مع سجلات منطقية فردية ، بغض النظر عن معامل التجميع ( انظر القسم الثامن من هذا الفصل ) . وتوجد ثلاثة إمكانيات :

• **الملف غير مجمع :** فى هذه الحالة .. يحذف BLOCK CONTAINS من FD .

```
FD   WEEKLY-PERSONNEL-REPORT
     RECORD CONTAINS 132 CHARACTERS
     LABEL RECORDS ARE OMITTED
     .
01   PERSONNEL-REPORT-RECORD . . .
```

حيث يسرى التجميع على الأوساط المغنطة فقط ؛ وإذا لاتكون ملفات البطاقات أو الطباعة مجمعة .

• **مجموعات ثابتة الطول :** يستخدم معها BLOCK CONTAINS integer-2 CHARACTERS

```
FD  MASTER-INVENTORY-FILE
    BLOCK CONTAINS 500 CHARACTERS
    RECORD CONTAINS 100 CHARACTERS
    LABEL RECORDS ARE STANDARD
```

أو يستخدم معها : BLOCK CONTAINS integer-2 RECORDS

```
FD  MASTER-INVENTORY-FILE
    BLOCK CONTAINS 5 RECORDS
    RECORD CONTAINS 100 CHARACTERS
    LABEL RECORDS ARE STANDARD
```

فى المثال الاول معامل التجميع المشمول هو :

٥٠٠ بايت / مجموعة
ــــــــــــــــــــ = ٥ سجلات / مجموعة
١٠٠ بايت / سجل

بينما فى المثال الثانى :

( ٥ سجل / مجموعة ) ( ١٠٠ بايت / سجل ) = ( ٥٠٠ بايت / مجموعة )

والصيغتان متكافئتان تماما .

• **مجموعات متغيرة الطول** : وتحدث عندما تختلف اطوال السجلات المنطقية الموجودة فى الملف، ويستخدم معها

```
BLOCK  CONTAINS integer-1 TO integer-2 CHARACTERS
RECORD  CONTAINS integer-3 TO integer-4 CHARACTERS
```

( انظر مثال ٥ – ٦ ) أو يستخدم معها

```
BLOCK  CONTAINS integer-2 RECORDS
RECORD CONTAINS integer-3 TO integer-4 CHARACTERS
```

( انظر مثال ٥ – ٧ )

مثـال ٥ – ٦ :

```
FD  MASTER-CUSTOMER-FILE
    BLOCK CONTAINS 220 TO 420 CHARACTERS
    RECORD CONTAINS 50 TO 100 CHARACTERS
    LABEL RECORDS ARE STANDARD
```

يحدد شكل ٥ – ٣ تكوين إحدى المجموعات متغيرة الطول ( بافتراض أن معامل التجميع هو ٣ ) وذلك باستخدام كوبل IBM OS/VS .

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| . . . | *gap* | BDW | RDW | Logical Record | RDW | Logical Record | RDW | Logical Record | *gap* | . . . |

BDW = كلمة واصف المجموعة block descriptor word ( يتم إدخالها تلقائيا عن طريق المترجم ) ، وتحدد طول المجموعة بما فى ذلك ٤ بايت خاصة بها .

RDW = كلمة واصف سجل record descriptor word ( يتم إدخالها تلقائيا عن طريق المترجم ) وتحدد طول السجل المنطقى التى تسبقه بما فى ذلك ٤ بايت خاصة بها .

شكل (٥ – ٣)

لا يشمل جزء RECORD CONTAINS كلمة RDW إلا أن BLOCK CONTAINS... CHARACTERS يشمل RDE, BDW وعلى ذلك.. فاقصر مجموعة لهذا الملف تحتوى على :

$$\frac{٢٢٠ - ٤}{٥٠ + ٤} = ٤ \text{ سجلات}$$

وتحتوى أطول مجموعة للملف على :

$$\frac{٤٢٠ - ٤}{١٠٠ + ٤} = ٤ \text{ سجلات}$$

وإيجازا فمعامل التجميع للملف هو ٤ .

مثال ٥ – ٧ :

```
FD  MASTER-CUSTOMER-FILE
    BLOCK CONTAINS 4 RECORDS
    RECORD CONTAINS 50 TO 100 CHARACTERS
    LABEL RECORDS ARE STANDARD
```

وهنا باعتبار مثال ٥ – ٦ فإن معامل التجميع يكون واضحا وهو ٤ . ويكون أقل طول مجموعة واقعى على ذلك هو:

٤ ( ٤ + ١٠٠ ) + ٤ = ٤٢٠ بايت

٤ ( ٤ + ٥٠ ) + ٤ = ٢٢٠ بايت

ومن الملاحظ هنا ان FD مكافىء تماما لنظيره فى مثال ٥ – ٦ .

## ٥ – ٤ وصف الملف وما يحتويه السجل

سبق تغطية تطبيق هذا الجزء على السجلات ثابتة الطول، والسجلات متغيرة الطول فى القسم السابق من هذا الفصل . وهذا الجزء اختيارى، لان المترجم يستطيع أن يحدد طول السجل من المستوى 01 الخاص بوصف السجل ( الذى يشمل حجم كل حقل ) . إلا إنه من المستحسن أستخدام RECORD CONTAINS دائما؛ حيث إنه يمثل توثيقا مفيدا للبرنامج كما أن بعض المترجمات تنتج رسالة تحذير ، إذا لم يتفق حجم الحقول مع قيمة هذا الجزء .

## أحتواء المجموعة أو السجل على صفر من الرموز

من الممكن فى كوبل IBM OS/VS كتابة :

```
BLOCK CONTAINS 0  [ CHARACTERS ]
                  [ RECORDS    ]
```

أو كتابة :

RECORD CONTAINS 0 CHARACTERS

وهذا هو اتساع extension لكوبل ANS النمطى ، الذى يتطلب عددا موجبا من الرموز او السجلات . ويحدد استخدام الصفر فى كوبل IBM حجم المجموعة أو السجل ، الذى يجب أن يؤخذ من عبارات لغة العمل التى تصف الملف أكثر ( انظر القسم الخامس من الفصل الرابع ) . وبالرغم من أن RECORD CONTAINS 0 CHARACTERS له أستخدامات عملية بسيطة فى الكوبل إلا أن تطبيقات BLOCK CONTAINS 0 يمكن ، بل يجب ، ان تستخدم فى جعل برنامج الكوبل مستقلا عن معامل التجميع ( بحيث إن تغيير معامل التجميع لا يحتاج إلى تغيير أو ترجمة برنامج مصدر الكوبل ) . (تذكر أن ملفات البطاقات والطباعة لايمكن أن يحدث فيها تجميع) .

**مثـال ٥ – ٨ :**

```
FD  OPEN-INVOICE-FILE
    BLOCK CONTAINS 0 CHARACTERS
    RECORD CONTAINS 250 CHARACTERS
    LABEL RECORDS ARE STANDARD
```

يؤخذ طول المجموعة من عبارة تحكم العمل المصاحبة للملف أثناء التنفيذ . فإذا كان الملف موجودا فعلا وله عنوان ملف .. فإنه يمكن أخذ طول المجموعة كذلك من عنوان الملف ( مع حذف الحاجة إلى تحديده فى عبارة لغة تحكم العمل ) .

## ٥ - ٥ وصف الملف وسجلات العناوين

تذكر من القسم رقم العاشر فى الفصل الأول ان لكل ملف موجود على وحدة اتصال مباشر - عنوان ملف file label (يقدمه نظام التشغيل تلقائيا ) والذى يحدد الاسم الذى يعرف به الملف لدى نظام التشغيل ، ونوع التنظيم ( تتابعى أو مفهرس أو نسبى) ، وموقع منطقة الفهرس ( إذا كان هناك فهرس ) ، وموقع سجلات البيانات، وحجم المجموعة ، ونوع السجل ( ثابت الطول أو متغير الطول ) وطول السجل ، وأى كلمات مرور لازمة للاتصال بالملف ، وطول الفترة التى يجب أن يحمى فيها الملف من حذفه .. الخ . وتجمع عناوين الملفات الموجودة على مجموعة أقراص مع بعضها البعض فى ملف خاص بنظام التشغيل يسمى بحجم جدول المحتويات volume table of contents (VTOC) أو دليل directory .

إن عناوين الملفات اختيارية بالنسبة لملفات الشرائط المغناطيسية ، ولكن بسبب مميزاتها العديدة .. فهى تستخدم دائما . وعناوين الملفات غير ممكنة لوحدات مدخلات ومخرجات الوحدة ( قارئات البطاقات والمثقبات والطابعات ) . الجزء LABEL RECORDS هو الجزء الوحيد المطلوب تواجده اجباريا فى FD . استخدم :

LABEL RECORDS ARE OMITTED

إذا كان الملف على وحدة مدخلات ، أو وحدة مخرجات ، أو إذا ما كان الملف على شريط مغناطيسى، وليس له عناوين ملف . وأستخدم :

LABEL RECORDS ARE STANDARD

إذا كان الملف فى وحدة اتصال مباشر ( قرص ، أو أسطوانة، أو قرص مرن .. إلخ ) أو إذا كان الملف على شريط مغناطيسى، وله عناوين ملف .

## ٥ - ٦ وصف الملف وسجلات البيانات

يسرد جزء DATA RECORDS الأسماء التى يعرفها المبرمج ، والتى يشار بها إلى السجلات فى جزء الإجراءات . ولهذا الجزء فائدة محددة : خاصة بالنسبة للسجلات ثابتة الطول ، حيث تتبع أسماء الكويل FD مباشرة ( فهى أول محتوى على المستوى 01 لوصف السجل ) . ويجب أن تتفق أسماء السجلات على المستوى 01 مع الأسماء الموجودة فى جزء DATA RECORDS . وجزء DATA RECORDS اختيارى دائما .

مثـال ٥ - ٩ :

```
FD  WEEKLY-TIME-FILE
    BLOCK CONTAINS 436 TO 1636 CHARACTERS
    RECORD CONTAINS 50 TO 200 CHARACTERS
    LABEL RECORDS ARE STANDARD
    DATA RECORDS ARE CLERICAL-TIME-RECORD
                     LAWYER-TIME-RECORD

        01 CLERICAL- TIME- RECORD •
```

( وصف سجل منطقى طوله ٥٠ بايت )

```
        01 LAWYER-  TIME- RECORD •
```

( وصف سجل منطقى طوله ٢٠٠ بايت )

بافتراض كوبل IBM OS/VS ، فان معامل التجميع لـ WEEKLY- TIME- FILE هو ٠٨ .

عادة ما يكون لملفات البطاقات ( غير مجمعة بالضرورة ) أكثر من نوع سجل واحد نظرا لان كل الحقول المطلوبة قد لاتكفيها بطاقة واحدة . وعندما يحدث ذلك .. فتقسم المعلومات عادة على عدة بطاقات وتكتب كل بطاقة من بطاقات المجموعة بشفرة لتحديد الحقول الموجودة فيها . وتوضع شفرة البطاقة card code عادة فى أول او فى آخر بايت من السجل (البطاقة)

مثـال ٥ - ١٠ :

من المرغوب فيه وضع معلومات عناوين بريدية فى بطاقات نمطية بها ٨٠ عموداً . ونظرا لأن البيانات تتطلب أكثر من ٨٠ رمزاً .. فإن الملف يحتوى على ثلاثة أنواع من السجلات . يمكن أن يكون وصف الملف FD ووصف السجل ( المستوى 01 ) لمثل هذا الملف على النحو التالى :

```
FD  MAILING-LABEL-FILE
    RECORD CONTAINS 80 CHARACTERS
    LABEL RECORDS ARE OMITTED
    DATA RECORDS ARE RECORD-TYPE-1-NAME
                     RECORD-TYPE-2-CITY
                     RECORD-TYPE-3-BUSINESS
```

```
01  RECORD-TYPE-1-NAME.
    05  CODE-1              PIC X.
    05  ID-1                PIC X(9).
    05  NAME-1              PIC X(30).
    05  STREET-1            PIC X(30).
    05  FILLER              PIC X(10).
01  RECORD-TYPE-2-CITY.
    05  CODE-2              PIC X.
    05  ID-2                PIC X(9).
    05  PHONE-2             PIC X(8).
    05  CITY-2              PIC X(30).
    05  STATE-2             PIC X(2).
    05  ZIP-2               PIC X(9).
    05  FILLER              PIC X(21).
01  RECORD-TYPE-3-BUSINESS.
    05  CODE-3              PIC X.
    05  ID-3                PIC X(9).
    05  BUSINESS-3          PIC X(70).
```

لاحظ ان جزء BLOCK CONTAINS محذوف لملف البطاقات غير المجمع . وقد استخدم جزء RECORD CONTAINS حتى يصدر الكمبيوتر رسالة تحذيرية إذا لم يكن مجموع خانات أى سجل على المستوى 01 مساوياً ٨٠ ( وهذا اختبار مفيد ، حيث انه لايوجد اتجاه لوجود سجلات متغيرة الطول ، ولكن توجد ثلاثة أنواع من السجلات طول كل منها ثابت وهو ٨٠ خانة ) . يحذف جزء LABEL RECORDS لملف البطاقات . يعمل جزء DATA RECORD كتوثيق للبرنامج ؛ ليسهل تحديد عدد الأنواع المختلفة من السجلات الموجودة فى الملف ، هل تستطيع تسمية الحقول فى كل نوع من أنواع السجلات المنطقية ؟

## ٥ - ٧ وصف الملف والخطية

يستخدم جزء الخطية LINAGE للفات المخرجات المطبوعة فقط وهو اختيارى دائما . وعند أستخدامه.. فإنه يعرف صفحة منطقية logical page ، تشمل هامشاً علوياً top margin ، وهامشاً سفلياً bottom margin وجسم للصفحة page body تكتب داخله الطباعة الفعلية . ويمكن أن تعرف منطقة فى أسفل جسم الصفحة بطريقة اختيارية، بإنها منطقة النهاية footing area .

يمكن استخدام الخطية LINAGE بدلاً من قنوات تحكم العربة ( القسم الرابع من الفصل الرابع ) ولايمكن استخدام الطريقتين لنفس الملف . ولايتاح جزء LINAGE مع كل المترجمات . انظر المشكلة رقم ٣٠ من الفصل الثامن للتعرف على أساليب التحكم فى الصفحات بدون LINAGE

مثـال ٥ - ١١ :

```
FD  SALES-BY-SALESPERSON-REPORT
    RECORD CONTAINS 132 CHARACTERS
    LABEL RECORDS ARE OMITTED
    LINAGE IS 54 LINES
        WITH FOOTING AT 44
        LINES AT TOP 6
        LINES AT BOTTOM 6
```

ملف الطباعة غير مجمع ، وليس له عنوان ملف . يعرف جزء الخطية LINAGE صفحة منطقية ، كما هو محدد في شكل ٥ – ٤.

وتحدث كل الطباعة داخل جسم الصفحة ( الجزء المظلل ) .

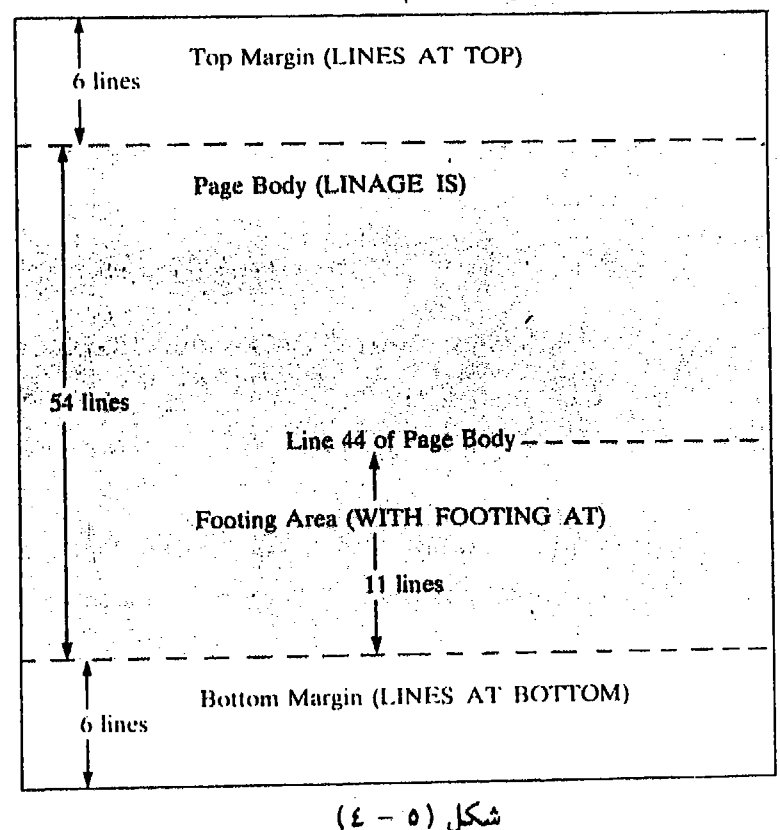
دائما يترك الهامشان العلوى والسفلى من الصفحة فارغين دائمًا ، وعلى هذا تخدم الهوامش فى فصل جسم الصفحة من جسم الصفحة التالى والسابق له . الجزءان :LINES AT TOP و LINES AT BOTTOM اختياريان إذا ما حذف أى منهما فيفترض فى الهامش المناظر له إنه لايحتوى على أى أسطر .

يمكن استخدام منطقة النهاية footing area داخل جسم الصفحة فى :

( ١ ) طباعة إجماليات الصفحة لحقول معينة .

( ٢ ) طباعة رسالة عندما يتقاطع جزء من التقرير مع حدود الصفحة، ويجب على هذا أن يستمر فى جسم الصفحة التالى.

( ٣ ) التأكد من وجود مكان كاف فى نهاية جسم الصفحة ، لأنهاء أى مادة تكون قد بدأت ( وعلى هذا حذف إمكانية استمرار شيء معين على الصفحات التالية ) . منطقة النهاية اختيارية وإذا ماحذف الجزء WITH FOOTING AT فيفترض أن منطقة النهاية تحتوى على صفر من الاسطر فى نهاية جسم الصفحة .




شكل (٥ – ٤)

تطبع فى جسم الصفحة أسطر التقرير الطبيعية . انظر الفصل السادس لمناقشة جزء الإجراءات عند استخدام عبارة WRITE فى إخراج ملف مع استخدام LINAGE .

**مثـال ٥ – ١٢ :**

```
•      FD  OUT-OF-STOCK-REPORT
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED
           LINAGE IS 66 LINES
```

يبلغ طول كل جسم صفحة ٦٦ سطرا ويجاور جسم الصفحة التالى مباشرة . منطقة النهاية بها صفر من الاسطر (أى إنها غير موجودة فى الصفحة) إلا أن اكتشاف منطقة نهاية سوف يحدد بالإشارة إلى عبارة WRITE عند الوصول إلى نهاية جسم الصفحة .

```
•      FD  SALES-BY-CUSTOMER-REPORT
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS OMITTED
           LINAGE 50 LINES
               LINES AT TOP SIZE-TOP-MARGIN
               LINES AT BOTTOM 5
```

أى من ، أو كل ، أجزاء LINAGE يمكن أن يحدد قيمة توضع فى تخزين قبل فتح OPEN الملف ( قبل إعداده للتشغيل ، انظر مثال مثال رقم ٥ من الفصل الثانى ) . فى هذه الحالة ، الرقم الذى ليس له إشارة الموجود فى موقع SIZE- TOP-MARGIN عند فتح OPEN الملف SALES- BY- CUSTOMER- REPORT يكون عبارة عن عرض الهامش العلوى . تسمح هذه السمة للبرنامج بإدخال او حساب عرض أى من مناطق الخطية LINAGE .

## ٥ – ٨ وصف السجل فى قسم الملفات

عندما يقوم برنامج هدف كوبل بتشغيل ملف .. فيجب أن تحجز منطقة من ذاكرة الكمبيوتر لاستقبال السجل الواقعى ( من ملف المدخلات )، أو لبناء السجل الواقعى ( الملف المخرجات ) ، وتسمى مثل هذه المنطقة بالذاكرة الاحتياطية buffer للملف معين.

والغرض من جزء وصف السجل هو وصف السجل المنطقى النشط حاليا، كما يظهر فى الذاكرة الاحتياطية . وفى كلمات أخرى ، يحدد جزء وصف السجل هذا الجزء من الذاكرة الاحتياطية الذى يمسك بالسجل المنطقى ، الذى يجرى عليه التشغيل حاليا .

**مثـال ٥ – ١٣ :**

لجزء برنامج مثال ٥ – ٩ الاستمرارية التالية :

```
FD  WEEKLY-TIME-FILE
    BLOCK CONTAINS 436 TO 1636 CHARACTERS
    RECORD CONTAINS 50 TO 200 CHARACTERS
    LABEL RECORDS ARE STANDARD
    DATA RECORDS ARE CLERICAL-TIME-RECORD
                     LAWYER-TIME-RECORD
```

• 01 CLERICAL- TIME- RECORD

(وصف سجل منطقى طوله ٥٠ بايت)

• 01 LAWYER- TIME- RECORD

(وصف سجل منطقى طوله ٢٠٠ بايت)

```
. . .
PROCEDURE DIVISION.
. . .
OPEN INPUT WEEKLY-TIME-FILE
. . .
READ WEEKLY-TIME-FILE RECORD
. . .
```

توضح عبارات جزء الإجراءات كيفية إعداد ملف المدخلات (OPEN INPUT WEEKLY- TIME- FILE) ، وكيفية الإدخال الفعلى لسجل من الملف READ WEEKLY- TIME- FILE RECORD . وتحقق عبارة READ مايلى:

( ١ ) إذا كانت الذاكرة الاحتياطية فارغة ، فإنها تدخل سجلا واقعيا، وتجعل أول سجل منطقى – فى هذا السجل الواقعى – متاحا للبرنامج .

( ٢ ) إذا كانت الذاكرة الاحتياطية تمسك حاليا بسجل واقعى، وتم إجراء تشغيل على كل سجلاته المنطقية .. فإنها تجرى نفس الإجراء مثل الحالة الأولى .

( ٣ ) إذا كانت الذاكرة الاحتياطية تمسك سجلا واقعيا، لم يتم تشغيل كل سجلاته المنطقية ، فإنها تجعل السجل المنطقى التالى ( الذى لم يجر عليه تشغيل ) متاحاً للبرنامج .

فى أى حالة من الحالات ، بعد عبارة READ ، يكون هناك سجلا منطقيا متاحا للبرنامج . ويرجع الأمر إلى المبرمج، ليتأكد ما إذا كان البرنامج يميز أى نوع من السجلات المنطقية . والإجراء العادى– هنا – هو حجز بايت معين فى كل نوع من أنواع السجلات كشفرة للسجل record code . وعلى هذا.. فأول بايت لكل سجلات الكتبة clerical يمكن أن يحتوى على الحرف C، بينما يمكن أن يحتوى أول بايت ، من كل السجلات للمحامين lawyer على الحرف L . أول السجل الذى له حرف C ،يجرى عليه تشغيل C بواسطة البرنامج ، كما أن البرنامج يجرى تشغيل L على السجل الذى له الحرف L .

## ارقـــام المســـتويات

كجزء أساسى من وصف السجل .. تحدد أرقام مستويات للحقول الموجودة فى السجل المنطقى ، وتتراوح من 01 الى 49 . والمستوى 01 محجوز لاسم السجل نفسه . أذا ازداد رقم المستوى عند الانتقال من حقل لآخر .. فيقال أن الحقل الثانى هو حقل جزئى subfield من الحقل الأول .

**مثـال ٥ – ١٤ :**

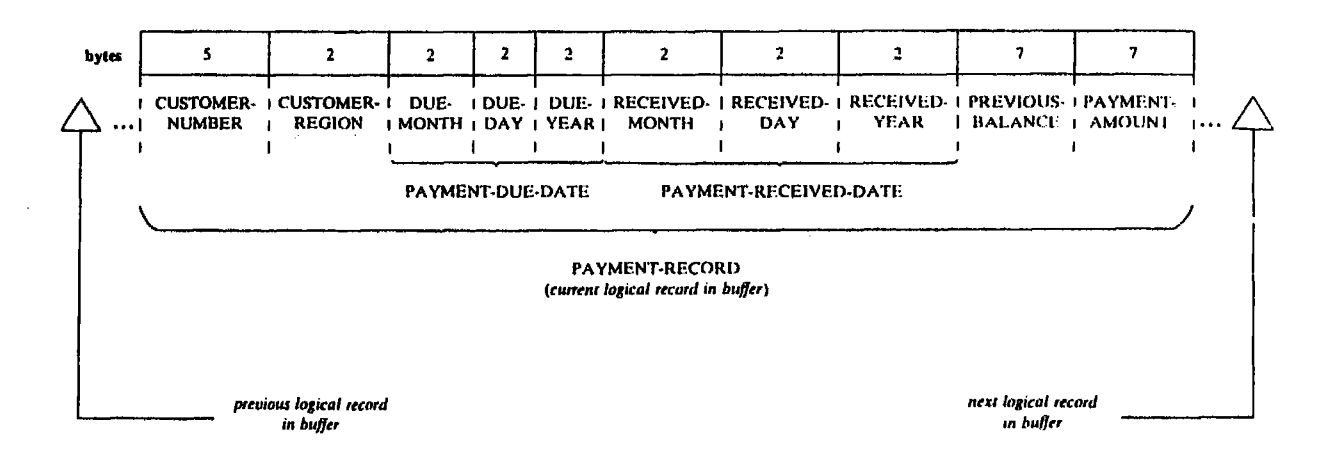
```
FD  PAYMENT-FILE
    BLOCK CONTAINS 100 RECORDS
    RECORD CONTAINS 33 CHARACTERS
    LABEL RECORDS ARE STANDARD

01  PAYMENT-RECORD.
    05  CUSTOMER-NUMBER         PIC X(5).
    05  CUSTOMER-REGION         PIC X(2).
    05  PAYMENT-DUE-DATE.
```

```
        10 DUE-MONTH              PIC X(2).
        10 DUE-DAY                PIC X(2).
        10 DUE-YEAR               PIC X(2).
    05 PAYMENT-RECEIVED-DATE.
        10 RECEIVED-MONTH         PIC X(2).
        10 RECEIVED-DAY           PIC X(2).
        10 RECEIVED-YEAR          PIC X(2).
    05 PREVIOUS-BALANCE           PIC S9(5)V99.
    05 PAYMENT-AMOUNT             PIC S9(5)V99.
```

يحدد رقم المستوى 01 المكتوب فى المنطقة A ، بداية جزء وصف السجل . وتكتب أرقام المستويات الاخرى ، وهى 05، 10 هنا، فى المنطقة B . لاحظ استخدام الترحيل فى جزء البيانات وذلك بالرغم من أن الترحيل غير مطلوب بالنسبة للمترجم ، إلا إنه يمثل نمطية كتابة شفرة مهمة جدا والتى توضح تكوين السجل .

الهيكل المعرف بواسطة ارقام مستويات المثال الحالى مرسوم فى شكل ٥ – ٥ .




شكل (٥ ـ ٥)

تسمى الحقول التى تشتمل على حقول جزئية مجموعة عناصر group items ، والحقول التى لاتشتمل على حقول جزئية تسمى عناصر فردية elementary items .

**مثـال ٥ – ١٥ :**

**group items**
PAYMENT-RECORD
PAYMENT-DUE-DATE
PAYMENT-RECEIVED-DATE

**elementary items**
CUSTOMER-NUMBER
CUSTOMER-REGION
DUE-MONTH
DUE-DAY
DUE-YEAR
RECEIVED-MONTH
RECEIVED-DAY
RECEIVED-YEAR

لاحظ أن العناصر الفردية لها جزء PIC يصف طول ونوع البيانات للعنصر . وليس لمجموعة العناصر مثل هذا الوصف؛ لإنها معرفة تماما بواسطة العناصر الفردية التى تتكون منها . وتشير الاسماء التى يعرفها المبرمج دائما إلى مجموعات العناصر والعناصر الفردية إلى سجل منطقى نشط حاليا فى الذاكرة الاحتياطية.

وكتابة الشفرة النمطية التى يوصى بها، هى استخدام اختصار لاسم السجل ( الاسم الموجود على مستوى 01 ) أو اختصار لاسم الملف ( الاسم الموجود على مستوى FD ) كسابقة لكل اسم بيانات موجود فى السجل . على هذا يصبح من الواضح إلى أى ملف أو سجل ينتمى عنصر بيانات معين .

**مثـال ٥ – ١٦ :**

```
FD  MAILING-LABEL-FILE
    RECORD CONTAINS 76
    LABEL RECORDS OMITTED
    DATA RECORDS MAILING-LABEL-NAME-STREET-REC
                 MAILING-LABEL-CITY-STATE-REC

01  MAILING-LABEL-NAME-STREET-REC.

    05 MAILING-LABEL-NAME-ST-CODE           PIC X.
    05 MAILING-LABEL-NAME-ST-ID             PIC X(5).
    05 MAILING-LABEL-NAME-ST-NAME           PIC X(30).
    05 MAILING-LABEL-NAME-ST-STREET         PIC X(30).
    05 FILLER                               PIC X(10).

01  MAILING-LABEL-CITY-STATE-REC.

    05 MAILING-LABEL-CITY-STATE-CODE        PIC X.
    05 MAILING-LABEL-CITY-STATE-ID          PIC X(5).
    05 MAILING-LABEL-CITY-STATE-PHONE       PIC X(8).
    05 MAILING-LABEL-CITY-STATE-CITY        PIC X(30).
    05 MAILING-LABEL-CITY-STATE-STATE       PIC X(2).
    05 MAILING-LABEL-CITY-STATE-ZIP         PIC X(9).
    05 FILLER                               PIC X(21).
```

هناك مستويان من السابقات، يستخدمان فعلا فى هذا المثال ، وهما:

( ١ ) كل اسم سجل، واسم حقل يبدأ بـ MAILING- LABEL ، وهو اختصار لاسم الملف MAILING- LABEL- FILE . أى اسم بيانات يبدأ بهذه السابقة.. يكون معروفا على هذا بأنه مصاحب للملف MAILING- LABEL- FILE . وهذا .. يجعل من الممكن قراءة جزء الإجراءات دون الإشارة المستمرة إلى جزء البيانات، لمعرفة أى سجل أو ملف يحتوى على حقل معين . (عندما يكون هناك اكثر من مستوى واحد 01 لنفس مستوى FD .. يقال عن أوصاف السجلات المنطقية إنها تعيد تعريف نفسها ضمنيا).

( ٢ ) يحتوى اسم كل حقل فى السجل MAILING- LABEL- NAME- STREET- REC على السابقة الثانية -NAME ST ، وهى اختصار NAME- STREET- REC ( مثال ذلك MAILING- LABEL- NAME- ST- CODE يصاحب MAILING- LABEL- FILE . وبصفة خاصة .. هناك حقلان من السجل -MAILING- LABEL- NAME STREET- REC فى هذا الملف ) . وبالمثل.. لكل حقل فى السجل MAILING- LABEL- CITY- STATE- REC سابقة ثانية ، وهى CITY- STATE فى اسمه.

**مثـال ٥ – ١٧ :**

يوجد واصفان لمستويات 01 فى مثال ٥ – ١٦ للسجل المنطقى النشط حاليا فى الذاكرة الاحتياطية ، وهذان الوصفان موضحان فى شكل ٥ – ٦ .

لاحظ أن كلاً من واصفى السجل متفقان مع تفسير أول ٦ بايت من السجل المنطقى . وكما يشير -MAILING- LABEL NAME- ST- CODE و MAILING- LABEL- CITY- STATE- CODE إلى نفس موقع الذاكرة ( بالاسم ) أول بايت من السجل المنطقى ) ، فيمكن أن يعالج اسما البيانات هذين كمرادفين خلال بقية البرنامج . وبالمثل.. يشير -MAILING LABEL- NAME- ST- ID و MAILING- LABEL- CITY- STATE- ID إلى البايت من ٢ – ٦ فى السجل المنطقى .. وعلى هذا.. فتبادل استخدام اسمى البيانات فى جزء الإجراءات لايؤثر على تنفيذ البرنامج . يمكن استخدام أول بايت من السجل المنطقى ( والتى يمكن الإشارة اليها بإنها MAILING- LABEL- NAME- ST- CODE أو بإنها -MAILING LABEL- CITY- STATE- CODE لتحتوى على شفرة سجل تعرفه كسجل اسم وشارع أو كسجل مدينة وولاية . ( انظر مثال رقم ١٣ والمشكلة رقم ٨٠ من هذا الفصل ).

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | NAME-ST-CODE | NAME-ST-ID | NAME-ST-NAME | | NAME-ST-STREET | | *unused* |
| bytes | 1 | 5 | 30 | | 30 | | 10 |
| ... | currently active logical record in buffer | | | | | | |
| bytes | 1 | 5 | 8 | 30 | 2 | 9 | 21 |
| | CITY-STATE-CODE | CITY-STATE-ID | CITY-STATE-PHONE | CITY-STATE-CITY | CITY-STATE-STATE | CITY-STATE-ZIP | *unused* |

شكل (٥ – ٦)

## الذاكرات الاحتياطية المتعددة

بالنسبة لملفات الاتصال التتابعى .. يمكن أن يكون هناك ميزة وجود ذاكرتين احتياطيتين ( أو أكثر ) : يجرى تشغيل على محتويات ، احداها داخل وحدة التشغيل المركزية، مع تغذية الثانية بالمجموعة الثانية من وحدة المدخلات والمخرجات فى نفس وقت التشغيل . وعدد الذاكرات الاحتياطية التقليدى لملفات الاتصال العشوائى فهو واحد . ويوجد جزء إضافى فى كوبل ANS فى عبارة SELECT ، يسمح للمبرمج بتحديد عدد الذاكرات الاحتياطية للملف كرقم صحيح .

```
SELECT . . . ASSIGN TO . . . [RESERVE integer {AREA  }] . . .
                                              {AREAS }
```

ربما يكون من الأفضل – إلا إذا كنت معتادا جدا على نظام الكمبيوتر، وتكون المدخلات والمخرجات – حذف جزء RE-SERVE ، وقبول العدد التقليدى للذاكرات الاحتياطية الذى يقدمه المترجم .

## ٥ - ٩ تكوين وصف البيانات

الغرض من محتوى وصف البيانات فى قسم الملفات هو الوصف التفصيلى لحقل السجل . يحدد شكل ٥ – ٧ التكوين العام لمحتوى وصف البيانات

```
    level-number {data-name-1}
                 {FILLER     }
(1)     [REDEFINES data-name-2]
(2)     [{PICTURE} IS character-string]
        [{PIC    }                    ]
                        {DISPLAY         }
                        {COMPUTATIONAL   }
(3)     [[USAGE IS]     {COMP            }]
                        {COMPUTATIONAL-3 }
                        {COMP-3          }
(4)     [BLANK WHEN ZERO]
(5)     [{JUSTIFIED} RIGHT]
        [{JUST     }      ]
(6)     [OCCURS integer TIMES]
(7)     [[SIGN IS]{LEADING }[SEPARATE CHARACTER]]
                  {TRAILING}
(8)     [{SYNCHRONIZED}{LEFT }]
        [{SYNC        }{RIGHT}]
(9)
```

شكل (٥ - ٧)

يجب أن يقع رقم المستوى بين 01 و 49 ، حيث يحجز الرقم 01 لأسماء السجلات . Data- name-1 هو أسم يعرضه المبرمج للحقل، لاستخدامه فى عبارات جزء الإجراءات التى تجرى تشغيلا على الحقل . فإذا كان موجودا احد الحقول ولن يجرى البرنامج تشغيلا عليه ، فيمكن استخدام كلمة FILLER من مكان اسم الحقل . ويمكن استخدام FILLER كلما كانت هناك حاجة لذلك .

**مثـال ٥ - ١٨ :**

```
FD  PAYMENT-FILE
    LABEL RECORDS ARE STANDARD.
01  PAYMENT-RECORD.
    05 CUSTOMER-NAME              PIC X(20).
    05 PAYMENT-AMOUNT             PIC S9(5)V99.
FD  WEEKLY-RECEIPTS-REPORT
    LABEL RECORDS ARE OMITTED.
01  WEEKLY-RECEIPTS-RECORD.
    05 CUSTOMER-NAME              PIC X(20).
    05 FILLER                     PIC X(5).
    05 PAYMENT-AMOUNT             PIC ZZ,ZZZ.99-.
    05 FILLER                     PIC X(65).
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
PROCEDURE DIVISION.
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    READ PAYMENT-FILE RECORD
    MOVE CUSTOMER-NAME OF PAYMENT-RECORD
       TO CUSTOMER-NAME OF WEEKLY-RECEIPTS-RECORD
    MOVE PAYMENT-AMOUNT OF PAYMENT-RECORD
       TO PAYMENT-AMOUNT OF WEEKLY-RECEIPTS-RECORD
```

تسمح قواعد لغة الكوبل بازدواج أسماء البيانات : فقد ظهر كل من CUSTOMER- NAME و PAYMENT- AMOUNT فى سجلين مختلفين فى هذا المثال . عندما يتكرر استخدام اسم بيانات فى جزء الإجراءات فيجب ان يؤهل qual- ified أى يجب أن يكون تابعا لمجموعة عناصر فريدة، مثل :

PAYMENT- AMOUNT OF PAYMENT CUSTOMER- NAME OF WEEKLY- RECEIPTS- RECORD,

ونوصى باستخدام أسماء بيانات فردية مع استخدام سابقات من أسماء الملفات أو السجلات ( مثال ٥ - ١٦ ) بدلا من استخدام تكرار الأسماء وتأهيل هذه الأسماء .

أذا استخدم جزء REDEFINES فيجب أن يتبع رقم المستوى مباشرة ، أما بقية الأجزاء فيمكن أن تظهر بأى ترتيب . ونناقش فى الأقسام التالية من ١٠ إلى ١٧ هذه الأجزاء بترتيب الأكثر استخداما أولا، فالأقل استخداما .(باستثناء موقع RE- DEFINES فهذا له الترتيب الموضح فى شكل ٥ - ٧ .

## ٥ - ١٠ جـــزء الصـــورة

يصف هذا الجزء شكل العنصر الفردى ، ولايمكن تحديده لمجموعة عناصر . ويمكن أن تشمل سلسلة الرموز أيًا من رموز الصورة picture characters التالية :

A B P S V X Z 0 9 / , . + – CR DB * $

تحدد سلسلة الرموز (١) عدد مواقع الرموز فى العنصر الفردى (٢) ونوع البيانات ( عددى أو حرفى عددى أو حرفى) الذى يشغل مواقع الرموز (٣) ، وما إذا كان عنصر البيانات يجرى عليه تنقيح يبدو فى صورة مرئية جيدة عند طباعته او عرضه على الشاشة أم لا .

وجزء PICTURE معقد لوجود عديد من البدائل . ونناقش كل رمز فردى لعموده فيمايلى . تذكر أنه يجب أن يكون لكل عنصر فردى جزء صورة PICTURE .

### الرمـــــز A

يمثل كل رمز A فى سلسلة الرموز موقعا ( بايت ) حرفيًا ، يمكن أن يحتوى على حرف أبجدى أو فرا غ فقط .

**مثـــال ٥ – ١٩ :**

```
01  EMPLOYEE-NAME.
    05  EMPLOYEE-LAST-NAME         PIC  A(15).
    05  EMPLOYEE-FIRST-NAME        PIC  A(10).
    05  EMPLOYEE-MIDDLE-INITIAL    PIC  A.
```

**تعليقات :** (١) لاتوجد صورة لمجموعة العناصر EMPLOYEE- NAME .

(٢) لكل عنصر فردى جزء PICTURE وصفى .

(٣) يبسط استخدام معامل التكرار من كتابة PICTURE :

A ( 10 ) هى اختصار لكتابة A عشر مرات .

A ( 4 ) هى اختصار لكتابة A أربعة مرات .

(٤) يشغل EMPLOYEE- LAST- NAME عدد ١٥ موقعا من مواقع الرموز ( بايت )، ويشغل EMPLOY- EE- FIRST- NAME خمسة مواقع للرموز بينما يشغل EMPLOYEE- MIDDLE- INITIAL موقعا واحدا فقط .

(٥) سوف تحتوى عناصر البيانات الثلاث كلها على حروف هجائية أو فراغات فقط .

(٦) استخدام PIC بدلا من DICTURE أمراً عادياً .

## الرمـــز B ( رمز تنقيح )

يمثل كل رمز B فى سلسلة PICTURE بايت واحداً يدخل فيه فراغ عندما تنقل البيانات إلى الحقل . والرمز B هو مثال لرمز تنقيح editing character ، يتسبب فى إعادة ترتيب البيانات أو إعادة تشكيلها عندما تنقل فى حقل صورة تنقيح .

**مثـال ٥ – ٢٠ :**

```
•       01   GREETING      PIC AABA(5).
```

**تعليقات :** (١) GREETING هو حقل مكون من ٨ بايت : ٢ منهما فراغات، أو حروف يتبعهما فراغ B يتبعه ٥ فراغات أو حروف A ( 5 ) .

(٢) ينتج عن MOVE "HITHERE" TO GREETING المحتويات التالية فى GREETING .

```
"HI THERE"
```

تتسبب B فى وجود فراغ بين أول حرفين ، وبقية الحروف الأخرى .

```
•       01  EMPLOYEE-INITIALS.
            05  COMPRESSED-INITIALS     PIC A(3).
            05  SEPARATED-INITIALS      PIC ABABAB.
```

**تعليقات :** (١) لايمكن تكرار مجموعة من رموز الصورة إلا بإعادة كتابتها ( AB (3) ينتج عنها ABBB، وليس ABABAB).

(٢) إذا ماالحتوت IF COMPRESSED- INITIALS على DAF فتضع

```
MOVE COMPRESSED-INITIALS TO
    SEPARATED-INITIALS
```

DAF فى SEPARATED- INITIALS .

## الرمـــز X

يمثل رمز الصورة الذى يستخدم بكثرة شديدة بايت يمكن أن يحتوى على أى رمز من مجموعة رموز الكمبيوتر ( مثل نظام EBCDIC لنظم IBM-370 ) . يستخدم عادة X فى جزء PICTURE الذى يجب أن يصاحب سجل على المستوى 01 ولايكون مقسما إلى حقول .

**مثال ٥ - ٢١ :**

```
        . . . . . . . . . . . . . . . . . . . . . . . . .
05  INPUT-SOCIAL-SECURITY-NUMBER      PIC X(9).
        . . . . . . . . . . . . . . . . . . . . . . . . .
05  PRINT-SOCIAL-SECURITY-NUMBER      PIC XXXBXXBXXXX.
        . . . . . . . . . . . . . . . . . . . . . . . . .
```

يمكن أن يحتوى INPUT- SOCIAL- SECURITY- NUMBER على قيمة، يتم إدخالها من قرص مثل "111223333"، رقم ضمان اجتماعى، لعامل بدون فراغات أو تنقيط غير ضرورى . وتحدد PIC X (9) مكانا كافيا لوضع أي ٩ رموز . وعندما يراد طباعة رقم الضمان الاجتماعى فإننا نريد تنقيح edit القيمة لجعلها سهلة القراءة . ويمكن عمل ذلك بإدخال فراغات عن طريق رمز الصورة B لإظهار الطباعة فى الصورة التالية : «111 22 3333».

وبصفة عامة .. عندما تستخدم بيانات عددية فى التعريف بدلا من الحسابات (مثل رقم الضمان الاجتماعى أو رقم الهاتف) فمن الأفضل استخدام رمز الصورة X .

## الرمـــز 9

يمثل كل رمز 9 فى الصورة موقعاً لرقم عشرى ( من صفر الى ٩ ) . على عكس الرمز A ، والرمز B ، والرمز X ، لايتطلب الرمز 9 بايت واحد دائما ( رمزا واحدا دائما ) من الذكرة . الحجم الفعلي لعنصر عددى ، والذى يعتمد على شفرة البيانات ، يعالج فى القسم رقم 18 من هذا الفصل . بينما لاتصبح برمجة الحسابات فى الكوبل باستخدام إعداد لها صورة X ، وذلك لأن الصورة 9 مصممة خصيصا للحسابات . لاتحدث PIC 9 أى تنقيح لعناصر البيانات العددية فهى لاتدخل بصفة - خاصة - تنقيطا أو تحذف أصفاراً ا رائدة leading zeros .

**مثـــال ٥ - ٢٢ :**

```
01  EMPLOYEE-INFORMATION-RECORD.
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    05  EMPLOYEE-INFO-SOCIAL-SECURITY        PIC X(9).
    05  EMPLOYEE-INFO-DEPENDENTS             PIC 99.
    05  EMPLOYEE-INFO-SICK-DAYS-LEFT         PIC 9(3).
    05  EMPLOYEE-INFO-VACATION-LEFT          PIC 9(3).
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
01  SAVINGS-ACCOUNT-RECORD.
    05  SAVINGS-ACCT-NUMBER-DEPOSITS         PIC 9(5).
    05  SAVINGS-ACCT-NUMBER-WITHDRAWS        PIC 9(5).
    05  SAVINGS-ACCT-NUMBER                  PIC X(7).
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

```
01   INVENTORY-MASTER-RECORD.
     05   INVENTORY-MAST-QUANTITY-ONHAND        PIC 9(7).
```

استخدم EMPLYEE- INFO- SOCIAL- SECURITY و SAVINGS- ACCT- NUMBER ، للتعريف ، وليس للحسابات ، وعلى هذا استخدمت صورة X . أما بقية الحقول الاخرى فيمكن أن تجرى عليها حسابات ، ولهذا اعطيت صورة 9. لاحظ إنه إذا نقلت 52 إلى حقل QUANTITY- ONHAND فإنها تخزن على النحو التالى 0000052 .

## الرمـــــز S

يحدد رمز S للصورة وجود إشارة عملية operational sign فى العدد . ولاتشغل إشارة العملية اى موقع تخزين إضافى؛ لانها تدخل فى التمثيل الداخلى للبيانات للرقم ( انظر القسم الحادى عشر من الفصل الأول والقسم رقم ١١ من هذا الفصل). لتمم الأزواج الثنائية Binary- twos- complement (COMP) ، والعشرى المضغوط (COMP-3) إشارات عمليات مبنية داخل تمثيل البيانات ، وعلى هذا بالنسبة إلى USAGE فإن حذف S من سلسلة الصورة يكافىء طلب القيمةالمطلقة absolute value . ويحجز تمثيل DISPLAY ( من نظام EBCDIC ونظام ASCII ) مكانا تلقائيا لإشارة العملية كجزء من تمثيل الرقم الموجود على أقصى اليمين، إلا أن استخدام هذا المكان أو عدمه يعتمد أساسا على ما إذا كانت هناك S موجودة فى PIC أم لا.

مثـــال ٥ – ٢٣ :

```
05   SECONDS-BEFORE-LAUNCH        PIC S9(5).
```

تعليقات : ( ١ ) إذا استخدمت S .. فيجب أن تكون أول رمز فى PICTURE .

( ٢ ) عند طباعة أعداد DISPLAY لها إشارات عمليات ، فيطبع الرقم الموجود على اقصى اليمين كحرف أو كفراغ . وعلى هذا ففى تمثيل EBCDIC ( جدول ٥ – ١ ) إذا طبعت SECONDS- BEFORE- LAUNCH الرقم 17 ثانية قبل الهبوط .. فإنها تطبع على النحو التالى : 00016 بينما إذا ماطبعت 12 ثانية بعد الهبوط فإنها تطبع على النحو التالى 0001K:

| الرقم الموجود فى اقصى اليمين | يطبع مع إشارة + على النحو التالى | يطبع مع إشارة – على النحو التالى |
|---|---|---|
| 0 | فراغ | فراغ |
| 1 | A | J |
| 2 | B | K |
| 3 | C | L |
| 4 | D | M |

| | | |
|---|---|---|
| N | E | 5 |
| O | F | 6 |
| P | G | 7 |
| Q | H | 8 |
| R | I | 9 |

( ٣ ) يجب أن تنقل الأعداد التى لها إشارات عمليات إلى حقول تنقيح وتطبع فى صورتها المنقحة لتجنب صعوبة تفسير المخرجات المطبوعة .

**مثـــال ٥ – ٢٤ :**

```
05   TOTAL-BALANCE-DUE       PIC  9(5).
```

بدون S فى صورة العنصر العددى .. ينتج الكمبيوتر إشارة موجب دائما لهذا العدد معطيا القيمة المطلقة له . ولايتطلب هذا تعليمات لغة آلة إضافية فى كل مرة يعالج فيها العدد فقط بل يمكن كذلك ان يكون خطيرا ، فمثلا بافتراض أن الحقل -TOTAL BALANCE- DUE يحتوى على 00015 ؛ محددا أن العميل مدينا بمبلغ 15 دولار . إذا دفع العميل 20 دولارا وطرحت القيمة من TOTAL- BALANCE- DUE فيحدث مايلى : 00015 مطروحا منها 00020 تعطى 0005 - وحيث انه لاتوجد S فى الصورة فتخزن على انها 0005 + . وعلى هذا يبدو ان العميل مازال مدينا بمبلغ 5 دولارات بينما يكون دائنا بهذا المبلغ!

من الأفضل ، إذا لم تكن القيمة المطلقة هى المطلوبة ، تحديد الإشارة فى الصورة دائما .

```
05   TOTAL-BALANCE-DUE       PIC  S9(5).
```

## الرمـــز V

يستخدم رمز الصورة V فى تحديد وجود علامة عشرية مفترضة assumed decimal point فى عنصر عددى . ويستخدم بالاتصال مع رمز الصورة 9 ويمكن أن يظهر مرة واحدة فقط فى PICTURE . إذا ماحذف الرمز V يفترض أن العلامة العشرية تتبع الرقم الموجود فى اقصى اليمين من العدد . وحيث ان العلامة العشرية الفعلية لاتتواجد فى التمثيل الداخلى للعنصر ، فإنها لاتستهلك موقع تخزين إضافى فى الذاكرة . تستخدم العلامات العشرية المفترضة فى ضبط العناصر العددية لأغراض التنقيح أو الحسابات .

**مثـــال ٥ – ٢٥ :**

```
05   HOURS-WORKED-THIS-WEEK      PIC  S99V9.
05   TOTAL-HOURS-FOR-YEAR        PIC  S9(4).
```

يحتوى حاليا على $42_{\wedge}5$ + $0063_{\wedge}$+ على التوالى ( يكون جزء من الساعة معنويا بالمقارنة بالأسبوع ، ولكنه لايكون كذلك بالنسبة للسنة ) . إذا كان مطلوبا من الكمبيوتر أن يضيف ساعات الاسبوع إلى ساعات السنة ، فيجب أن يذكر له موقع العلامة العشرية .

| صحيح | غير صحيح |
|---|---|
| ساعات الاسبوع = $42_{\wedge}5$ | ساعات الاسبوع = 425 |
| ساعات السنة = $0603_{\wedge}$ | ساعات السنة = 0603 |
| المجموع $0645_{\wedge}5$ | المجموع 1028 |

إذا كانت النتيجة الصحيحة ستخزن فى اسم البيانات TOTAL- HOURS- FOR- YEAR .. فيجب أن يحذف الجزء الكسرى ( 5 ر ) ، أو يجب أن يقرب العدد . إذا حذف الكسر العشرى باستخدام PIC S9(4) فتكون النتيجة 0645+ ، ويضيف التقريب 1 إلى الرقم الموجود على أقصى اليمين لحفظه إذا كان الرقم الموجود أقصى اليسار الذى يحذف 5 أو أكثر . فإذا قربت $0645_{\wedge}5$ إلى PIC S9(4) فتصبح النتيجة 0646+ .

## الرمــــز Z

يستخدم رمز الصورة Z بدلا من الرمز 9 لتمثيل موقع الرقم العشرى المراد أستبداله بفراغ إذا ما احتوى هذا الموقع على صفر من الاصفار الرائدة . لايمكن استخدام الحقول المعرفة برموز تنقيح ( بما فى ذلك Z ) فى الحسابات ، وذلك بالرغم من إنها يمكنها أن تستقبل نتائج الحسابات .

**مثـــال ٥ – ٢٦ :**

- 05 PRICE PIC ZZZZ9.

إذا احتوىPRICE على القيمة 203 وكان جزءاً من سجل مخرجات على الطابع ، فإنه يطبع على النحو التالى bb 203 (b تعنى موقعا فارغا ) . أول رمزين Z فى PIC مناظرين لأصفار رائدة فى القيمة المطبوعة ، وعلى هذا يترك فراغ فى هذين الموقعين.

الرمز Z الرابع فى PIC يناظر 0 فى القيمة ، كذلك إلا إنه ليس صفرا رائدا ( أى إنه لايسبق كل الأرقام غير الصفرية). وعلى هذا .. يطبع الصفر ( إذا احتوى PRICE على أصفار فالقيمة المطبوعة تصبح bbbb0 حيث أن 9 تمثل صفر دائما سواء كان هذا الصفر رائدا أم لا ) .

- 05 PRICE PIC ZZZZ.

إذا احتوى PRICE على صفر، وطبعت قيمته فإنها تظهر على النحو التالى : bbbb .

## رمـز تنقيـح النقطـة

تمثل النقطة ( العلامة العشرية ) الموقع الفعلى للعلامة العشرية فى الحقل ( المناظرة لموقع V ، التى تمثل علامة عشرية إفتراضية فى الصورة ) . وتشغل النقطة "." بايت واحداً من الحقل فى الذاكرة وتظهر عند طباعة الحقل أو ظهوره على الشاشة . وتعتبر النقطة رمز تنقيح ، لذا فلا تستخدم الحقول الموجودة فى صورتها نقطة فى الحسابات ، بل يمنع كذلك ظهور النقطة كآخر رمز فى الصورة .

**مثـال ٥ – ٢٧ :**

```
01  INVOICE-DETAIL-LINE.
    05  INVOICE-PART-NUMBER          PIC  X(5).
    05  FILLER                       PIC  X(8).
    05  INVOICE-PART-DESCRIPTION     PIC  X(20).
    05  FILLER                       PIC  X(5).
    05  INVOICE-QUANTITY-PURCHASED   PIC  ZZ9.
    05  FILLER                       PIC  X(3).
    05  INVOICE-UNIT-PRICE           PIC  ZZZ.99.
    05  FILLER                       PIC  X(10).
    05  INVOICE-TOTAL-PRICE          PIC  ZZZZZZ.99.
```

**تعليقـات :**

( ١ ) مجموعة العناصر ( المستوى 01 ليس لها PICTURE ) .

( ٢ ) الحقل INVOICE- PART- NUMBER هو حقل عددى، يستخدم فى التعريف فقط ، وعلى هذا .. استخدمت X فى تعريفه .

( ٣ ) كلمة FILLER هى كلمة محجوزة، تستخدم فى موقع اسم بيانات لحقل لن يجرى عليه تشغيل فى جزء الإجراءات . وقد استخدمت هنا فى تسمية حقول فارغة تفصل عناصر بيانات فعلية عند طباعتها أو عرضها على شاشة .

( ٤ ) تم تنقيح INVOICE- QUANTITY- PURCHASED للطباعة . وهو حقل من 3 خانات ( 2 ممثلتين بالحرف Z خانتين والثالث بالرمز 9 ) ، وتكون أول خانتين فارغتين إذا احتويتا على أصفار .

( ٥ ) تم تنقيح INVOICE- TOTAL- PRICE, INVOICE- UNIT- PRICE للطباعة . ويشمل هذا وضع فراغات بدلا من الاصفار الرائدة وطباعة العلامة العشرية الفعلية . لاحظ استخدام 9 بدلا من Z بعد "." فى PIC . وهذا لتأكيد ان العلامة العشرية تطبع دائما ، وتكون رموز التنقيط المحاطة بالرمز Z فى PIC فراغات إذا ما كانت Z السابقة والتالية تمثل أصفارا رائدة .

## رمـز تنقيح الفاصلة

تمثل الفاصلة "," بايت واحداً يوضع فيه رمز الفاصلة . ومثل النقطة.. فظهور فاصلة بين اثنين Z تستبدلان بفراغ، يجعل الفاصلة تستبدل بفراغ كذلك . كما ان الفاصلة تستبدل بفراغ كذلك اذا ما وجدت فراغات فقط على يسارها . لايمكن استخدام الحقول العددية التى يوجد فاصلة فى الصور الخاصة بها فى الحسابات (وذلك بالرغم من أنها يمكن أن تستقبل حقولا عددية).

مثــال ٥ – ٢٨ :

| *PICTURE* | *Value* | *Prints As* |
|---|---|---|
| ZZ,ZZZ.99 | 00010‸05 | *bbbb*10.05 |
| ZZ,ZZZ.99 | 00123‸45 | *bbb*123.45 |
| ZZ,ZZZ.99 | 01000‸00 | *b*1,000.00 |
| ZZ,ZZZ.99 | 98765‸43 | 98,765.43 |
| ZZ,ZZZ.ZZ | 00000‸00 | *bbbbbbbbb* |
| ZZ,ZZZ.ZZ | 00000‸03 | *bbbbbbbb*3 (falsified) |
| ZZ,ZZZ.99 | 00000‸03 | *bbbbbb*.03 (correct version of above) |

# علامة الدولار كرمز تنقيح

يمكن أن تستخدم علامة الدولارفى الصورة لإدخال علامة دولار ثابتة fixed dollar sign ، تطبع على يسار أول موضع رقم مباشرة .

مثــال ٥ – ٢٩ :

| *PICTURE* | *Value* | *Prints As* |
|---|---|---|
| $Z,ZZZ.99 | 0072‸15 | $*bb*72.15 |
| $Z,ZZZ.99 | 9876‸54 | $9,876.54 |

يمكن أن تحدد علامة الدولار علامة دولار متحركة floating dollar sign تطبع على يسار أول رمز معنوى significant character مباشرة ( أول رمز معنوى هو أول رقم معنوى من العنصر العددى ، أو من الممكن أن يكون العلامة العشرية نفسها ) . بهذا الاستخدام يكون هناك حاجة لأكثر من علامة دولار واحدة فى PICTURE : تمثل العلامة الموجودة على أقصى اليسار موقعا لعلامة الدولار نفسها ، أما بقية علامات الدولار.. فتمثل مواقع أرقام فى الحقل .

مثــال ٥ – ٣٠ :

| | *PICTURE* | *Value* | *Prints As* |
|---|---|---|---|
| (*a*) | $$,$$$.99 | 0001‸23 | *bbbb*$1.23 |
| (*b*) | $$,$$$.99 | 0845‸79 | *bb*$845.79 |
| (*c*) | $$,$$$.99 | 9876‸54 | $9,876.54 |
| (*d*) | $$,$$$.99 | 12345‸67 | $2,345.67 |
| (*e*) | $$,$$$.99 | −0000‸05 | *bbbbb*$.05 |

**تعليقــات :**

من ( أ ) الى ( هـ ) .. تتحرك علامة الدولار إلى أعلى رقم فى القيمة المنقحة . ( د ) يوجد 6 مواقع أرقام فقط فى PIC-TURE ( 4 علامات دولار بعد علامة الدولار الأولى بالإضافة إلى 2 من رمز 9 ) . وعلى هذا .. القيمة المكونة من سبعة أرقام، يحذف منها رقم من ناحية اليسار .

( هـ ) لايوجد تحديد لإشارة جبرية فى PICTURE ، وتطبع على هذا القيمة كما لو كانت موجبة . وهنا أول رمز معنوى هو العلامة العشرية وتنتقل علامة الدولار إليها .

تذكر أن علامة الدولار المتحركة تتطلب علامة دولار إضافية فى PICTURE عن عدد مواقع الأرقام التى يجرى عليها التنقيح . ولايمكن استخدام العناصر التى لها علامة دولار فى PIC الخاصة بها .

## رموز تنقيح التحكم فى الإشارة CR , DB , + , -

تستخدم هذه الرموز أساسا فى تنقيح القيم العددية السالبة .

**مثــال ٥ - ٣١ :**

يمكن استخدام رموز الصورة DB أو (CR) فى الناحية اليمنى من صورة العدد، لتحديد قيمة المدين ( أو الدائن ) السالبة .

| | PICTURE | Value | Prints As |
|---|---|---|---|
| (*a*) | $Z,ZZZ.99CR | 0010,56 | $*bbb*10.56*bb* |
| (*b*) | $Z,ZZZ.99CR | −1435,72 | $1,435.72CR |
| (*c*) | $$,$$$.99BDB | −3476,52 | $3,476.52 DB |

**تعليقــات :**

( أ ) و ( ب ) اذا كانت القيمة سالبة.. فتطبع CR أو (DB) وإلا فإنها تستبدل بفراغات .

( جـ ) تتسبب B هنا فى وجود فراغ بين آخر رقم مطبوع ، و DB او (CR) .

رمزا تحكم الإشارة الأخران هما : + و- ويعملان بالتماثل ، إلا فى وجه واحد : تتسبب + فى طباعة إشارة العدد دائما سواء ( كان موجبا أو سالبا ) بينما تتسبب – فى إظهار الإشارة عندما تكون القيمة المنقحة سالبة فقط . ويمكن استخدام رموز الصورة هذه فى موقع رائد leading position ( يسبق أول موقع لرقم مباشرة ) او فى موقع متخلف trailing posititon (بعد موقع آخر رقم مباشرة ) . وتستخدمان كذلك فى حالة ثابتة او متحركة مثل علامة الدولار تماما .

**مثــال ٥ - ٣٢ :**

| | PICTURE | Value | Prints As |
|---|---|---|---|
| (*a*) | $$,$$$.99− | −2345,67 | $2,345.67− |
| (*b*) | $$,$$$.99− | 0023,78 | *bbb*$23.78*b* |
| (*c*) | ZZZ.9+ | −763,7 | 763.7− |

| | | | |
|---|---|---|---|
| (*d*) | −ZZ.9 | 23‸7 | *b*23.7 |
| (*e*) | −ZZ.9 | −00‸7 | −*bb*.7 |
| (*f*) | +ZZ9 | −005 | −*bb*5 |
| (*g*) | + +,+ + +.9 | 0003‸7 | *bbbb* + 3.7 |
| (*h*) | − −,− −9 | 0000 | *bbbbb*0 |
| (*i*) | − −;− −9 | −0100 | *bb* − 100 |
| (*j*) | + +,+ +9 | 0000 | *bbbb* + 0 |

**تعليقــات :**

( أ ) يتكرر استخدام إشارة السالب المتخلفة فى تقارير الأعمال لإظهار الأعداد السالبة .

( ب ) ترك موقع إشارة السالب المتخلفة فارغا للقيمة الموجبة .

( جـ) يحتوى موقع إشارة الموجب المتخلفة على إشارة القيمة .

( د ) حل محل إشارة سالب الرائدة الثابتة فراغ للقيمة غير السالبة .

( هـ) تطبع إشارة سالب الثابتة الرائدة دائما فى أول موقع .

( و ) أشارة الموجب الثابتة الرائدة تضع الإشارة فى أول موقع دائما .

( ر ) أشارة الموجب المتحركة تنتقل إلى أول رمز معنوى .

( ز ) أشارة السالب المتحركة تطبع إذا كانت القيمة سالبة فقط ، فيجبر إنهاء PIC بالرمز 9 ، 0 على طباعته .

( ع) تطبع إشارة السالب المتحركة عندما تكون القيمة سالبة .

( غ) الصفر أعتبر موجبا .

فى أعداد تقارير الأعمال ، عادة ما تعالج تحكمات الإشارة بإشارة سالب متخلفة أو برمز CD أو رمز DB متخلف .

وكما هو الحال مع رموز التنقيح الأخرى لايمكن استخدام الحقول الموجود فى PIC الخاص بها رموز تحكم فى الإشارة فى الحسابات .

## النجمة كرمز تنقيح

تشير حماية الشيكات check protection إلى عمليات ملأ مواقع الأصفار الرائدة فى العدد بنجوم، بحيث يمكن حماية القيمة المدونة فى الشيك من التزوير : 23.45 ***$ . وتنفذ حماية الشيكات برمز النجمة فى الصورة .

**مثــال ٥ – ٣٢ :**

| | *PICTURE* | *Value* | *Prints As* |
|---|---|---|---|
| (*a*) | $*,***.99 | 12345‸67 | $2,345.67 |
| (*b*) | $*,***,***.99 | 0001234‸56 | $****1,234.56 |
| (*c*) | $*,***.** | 0000‸00 | $*****.** |

تعليقـــات :

( أ ) يمكن لعدد 4 نجوم و 2 من الرمز 9 أن يتسعوا لستة أرقام ، ويحدث حذف من ناحية اليسار .

( ب ) استبدلت الاصفار الرائدة والفواصل الرائدة بنجوم ، وتبقى الأرقام المعنوية والفواصل المعنوية .

(جـ) تطبع العلامة العشرية دائما عندما تستخدم حماية الشيكات حتى اذا كانت القيمة مساوية للصفر .

## الصفر كرمز تنقيح

يستخدم رمز الصورة 0 فى حجز موقع داخل النتيجة المنقحة، يطبع فيه صفر .

**مثـال ٥ – ٣٤ :**

احد الاستخدامات التقليدية للصفر كرمز تنقيح ، هو تقديم صفرمن السنتات بحيث يمكن طباعة العناصر – التى تحفظ كدولارات فى ذاكرة الكمبيوتر – كدولارات وسنتات .

| *PICTURE* | *Value* | *Prints As* |
|---|---|---|
| $ZZ,ZZZ.00 | 01205 | $*b*1,205.00 |
| $ZZ,ZZZ.00 | 00002 | $*bbbbb*2.00 |
| $$$,$$$.00 | 00002 | *bbbbb*$2.00 |

## الشرطة المائلة كرمز تنقيح

يستخدم رمز الصورة " / " فى حفظ بايت فى النتيجة المنقحة يمسك دائما بشرطة مائلة . تفيد الشرطة المائلة خاصة عند تنقيح إعداد تمثيل تواريخ ( مثل يمكن ان ينقح 102383 ليأخذ الشكل 10/23/83 باستخدام PIC 99/99/99 .

## رمــز التنقيـــح P

تستخدم سلسلة من الرمز P لتحديد موقع علامة عشرية افتراضية ، عندما يقع خارج مدى الأرقام المحفوظة فى الذاكرة . لايعد P رمز تنقيح، ويستخدم فى تعريف حقول عددية تستخدم فى الحسابات . ولا يشغل P أى موقع تخزين أضافى فى الحقل، كما أنه موقع يحدد موقع علامة عشرية افتراضية . ويمكن أن يظهر P فى النهاية اليسرى فقط او النهاية اليمنى فقط من الحقل العددى . والرموز الوحيدة التى يمكن أن تسبق P على اليسار هى S , V ، والرموز الوحيدة التى يمكن ان تتبع P على اليمين هى V .

**مثـال ٥ – ٣٥ :**

| | *PICTURE* | *Value in Memory* | *Value Used in Computation* |
|---|---|---|---|
| (*a*) | PP99 | 12 | ‸0012 |
| (*b*) | 99PP | 12 | 1200‸ |
| (*c*) | PP99PP | — | — |
| (*d*) | SPPP999 | 735 | +‸000735 |
| (*e*) | SVPPP999 | 735 | +‸000735 |
| (*f*) | SPP999 | 1234 | +‸00234 |
| (*g*) | S9(3)P(4) | 538 | +5380000‸ |
| (*h*) | SP99 | 03 | +‸003 |

تعليقـــات :

( أ ) و ( ب ) و ( ر ) تقع العلامة العشرية الافتراضية بعد عدد المواقع التى تمثلها P من ناحية اليسار .

( جـ) غير صحيح : لايمكن أن تظهر P فى كل من النهايتين .

( د ) S يمكن أن تسبق P من ناحية اليسار .

( هـ) V زائدة هنا ، لأن العلامة العشرية الافتراضية تم تثبيتها فعلا على يسار P الموجودة فى أقصى اليسار .

( و ) إذا كانت القيمة فى الذاكرة أرقام أكثر عن عدد رمز 9 فى PIC .. تحذف أرقام من ناحية اليسار ، وتحدد العلامة العشرية الافتراضية فى القيمة المحذوفة بواسطة P .

( ر ) لايعنى شيئا لعمل P ان الرقم الموجود على أقصى اليسار صفر .

## ٥ - ١١ جـــزء الاســـتخدام (شكل ٥ . ٧ - السطر رقم 3)

عادة مايحذف الجزء الذى يحدد التمثيل الداخلى المستخدم لعنصر البيانات ، من وصف البيانات حيث إن استخدام DIS-PLAY التقليدى ، هو الأكثر مناسبة فى استخدامه .

## الاستخدام هو للعرض

تحت DISPLAY يمثل عنصر البيانات بشفرة رموز نظام الكمبيوتر ( سواء كانت EBCDIC أم ASCII )، والتى يمثل فيها الرمز الحرفى عددى بشفرة ثنائية تحتاج إلى بايت واحد من الذاكرة . تكون DISPLAY إجبارية إذا كان عنصر البيانات ( ١ ) حرفيًا عدديًا PIC X ( ٢ ) حرفيًا PIC A ( ٣ ) أى نوع الحقل فى سجل ملف بطاقة ( سواء كان مدخلات أو مخرجات). ( ٤ ) أى نوع الحقل فى سجل مخرجات على طابع ( ٥ ) أى نوع الحقل يكون مدخلات أم مخرجات على شاشة أو نهاية طرفية تنتج نسخا دائمة ( ٦ ) حقلاً عدديًا تشمل صورته رمزاً واحدا أو أكثر من رموز التنقيح ( ٧ ) أى نوع من حقل المدخلات من ملف كان DISPIAY فعلا . ويوصى باستخدام DISPLAY إذا كان عنصر البيانات حقلا عدديا مستخدما فى التعريف ، وليس فى الحسابات ( يمكن أن يكون فى مخزن العمل أو فى أى نوع من أنواع وحدات التخزين المساعد للمدخلات والمخرجات).

**مثـــال ٥ - ٣٦ :**

بعد كل حقل مكتوب الأسباب التى من أجلها يجب أن تظهر DISPLAY باستخدام الأرقام المذكورة عاليه .

```
01  INPUT-TIME-CARD-RECORD.
    05  TIME-CARD-ID        PIC X(5)     USAGE IS DISPLAY.   (1, 3, 7)
    05  TIME-CARD-NAME      PIC A(20)    USAGE DISPLAY.      (2, 3, 7)
    05  TIME-CARD-HOURS     PIC S9(2)V9.                     (3, 7)
```

## الاستخدام يكون للحسابات (COMP)

COMP هو الاستخدام النمطى لكويل ANS للعناصر العددية التى تستخدم فى الحسابات . ويتغير تمثيل البيانات الدقيقة المستخدم لبيانات COMP من كمبيوتر لآخر ( وعادة ماتكون أحدى صيغ مكمل الأزواج الثنائية -binary- twos

complement . وبالرغم من أن الكمبيوتر يمكن أن يؤدى حسابات على أعداد COMP بكفاءة أعلى من أعداد DISPLAY ، إلا إنه يوصى بالنسبة للمبرمج الجديد أن يتذكر مايلى : لاتستخدم COMP إذا كان عنصر البيانات ( ١ ) محتملاً DISPLAY (٢) غير عددى أو إنه منقحاً ( ٣ ) حقلاً مصاحباً لوحدة سجل وحدة أو نهاية لانتاج نسخة دائمة أو شاشة ( ٤ ) عدديًا ، ولكنه مستخدم فى التعريف فقط ( استخدام PIC X DISPLAY بدلا من ذلك ) .

استخدم COMP إذا كان عنصر البيانات ( أ ) حقلا عدديا فى سجل مدخلات من ملف شريط أو قرصاً وهو COMP فعلا ( ب ) حقلا عدديا مستخدما فى الحسابات وهو حقل فى سجل يكون مخرجات فى ملف شريط او قرصاً يصبح بعد ذلك مدخلات ويستخدم فى حسابات لاحقة ( جـ) فى مخزن العمل ( ليس حقل سجل ملف ) ويكون عدديا غير منقح، ويستخدم فى الحسابات .

**مثـــال ٥ - ٣٧ :**

```
01  DISK-OUTPUT-RECORD.
    05  DISK-CUSTOMER-NUMBER   PIC  X(6).
    05  DISK-CUSTOMER-NAME     PIC  X(30).
    05  DISK-LAWYER-NUMBER     PIC  X(3).
    05  DISK-JOB-NUMBER        PIC  X(4).
    05  DISK-JOB-HOURS         PIC  S9(3)V9     COMP.
    05  DISK-JOB-RATE          PIC  S9(3)V99    COMP.
```

رقم العميل، ورقم المحامى، ورقم العمل... كلها أرقام مستخدمة فى التعريف فقط ، وعلى هذا فهى معرفة باستخدام PIC X DISPLAY ( تذكر أن DISPLAY هو الخيار التقليدى ) . رقم العميل من النوع الحرفى عددى ، ومن ثم فلابد أن يوصف على هيئة PIC X DISPLAY . عدد ساعات العمل ومعدل الأجر فى الساعة هما حقلان عدديان يجرى عليهما حسابات . وعلى هذا .. فمن الكفاءة الاكثر تخزينها على قرص فى صورة COMP . وفيما بعد.. يصبحان مدخلات فى عناصر بيانات COMP ، ويستخدمان فى حساب قيمة فاتورة العميل .. إلخ .

**مثـــال ٥ - ٣٨ :**

```
. . . . . . . . . . . . . . . . . .
05  ORIGINAL-PRICE       PIC  S9(3)V99.
05  DISCOUNT             PIC  S9(3)V99.
05  DISCOUNTED-PRICE     PIC  S9(3)V99.
. . . . . . . . . . . . . . . . . .
PROCEDURE  DIVISION.
. . . . . . . . . . . . . . . . . .
    SUBTRACT  DISCOUNT  FROM  ORIGINAL-PRICE  GIVING
        DISCOUNTED-PRICE
```

حيث أن كل عناصر البيانات الثلاث من نوع DISPLAY فإن المترجم يدرك ضرورة تحويلها إلى COMP أو (COMP-3) بغرض أداء الحسابات . وبعد ذلك تحول النتائج مرة اخرى الى DISPLAY . ويمكن أن تكون الترجمة التقليدية للغة الآلة لعملية الطرح كما يلى :

( ١ ) حول ORIGINAL- PRICE إلى صيغة COMP ، وضع النتيجة فى موقع مؤقت من الذاكرة T1 .

( ٢ ) حول DISCOUNT الى صيغة COMP وضع النتيجة فى موقع مؤقت من الذاكرة T2 .

( ٣ ) أطرح محتويات موقع الذاكرة T2 من محتويات موقع الذاكرة T1 ، وضع الفرق فى موقع مؤقت من الذاكرة T3 .

( ٤ ) حول محتويات موقع الذاكرة المؤقت T3 إلى صيغة DISPLAY ، وضع النتيجة فى DISCOUNTED- PRICE .

إذا عرف عناصر البيانات الثلاثة بأنها من نوع USAGE COMP .. فيمكن أن تصبح الترجمة اقتصادية أكثر :

( ١ ) أطرح محتويات DICOUNT من محتويات ORIGINAL- PRICE وضع الفرق فى الموقع DISCOUNTED- PRICE

## الاستخدام من نوع الحسابات (COMP-3)

COMP-3 هو أتساع لكوبل ANS المتاح على بعض ، وليس كل ، نظم الكمبيوتر ، وهو يشبه COMP فى قدرة وحدة التشغيل المركزية على عمل حسابات بكل من أعداد COMP-3 , COMP .

كما أن كلاً من الشفرتين تحتاج إلى ذاكرة أقل من تمثيل قيمة عددية معينة، مستخدمة فى حسابات من أنواع حسابات الأعمال ( أفحص دليل المورد المتاح لديك ) . استثناء : لاتستخدم COMP-3 للادلة subscripts ( انظر الفصل العاشر ) .

**مثـال ٥ - ٣٩ :**

استبدال - مثال ٥ - ٣٨ - PIC S9 (3) V99 بـ PIC S9 (3) V99 COMP-3 يعطى برنامج هدف أكثر كفاءة على كمبيوتر من نوع IBM-370 . استخدام COMP لعناصر البيانات الثلاثة يعطى ترجمة اقل كفاءة مباشرة اما استخدام DI-SPLAY ( كما فى مثال ٥ - ٣٨ ) فيعطى أقل كفاءة ترجمة .

## ٥ -١٢ جـــزء الفـــراغ (شكل ٥ . ٧ - السطر رقم 4)

يتسبب جزء BLANK WHEN ZERO فى وضع أماكن فارغة فى عنصر البيانات عندما ينقل إليه أصفارا . يمكن استخدام هذا الجزء مع العناصر العددية التى لايستخدم معها حماية للشيكات فقط . ويعتبر عنصر البيانات الذى يصاحبه BLANK WHEN ZERO ، بانه عنصر تنقيح تلقائى، وعلى هذا .. يجب أن يكون USAGE له من نوع DISPLAY .

**مثـال ٥ - ٤٠ :**

```
01  ACCOUNTS-RECEIVABLE-LINE.
    05  A-R-CUSTOMER-NAME      PIC  X(30).
    05  FILLER                 PIC  X(5).
    05  A-R-TYPE-PAYMENT       PIC  XBX.
    05  FILLER                 PIC  X(4).
    05  A-R-DAYS-PAST-DUE      PIC  ZZ9         BLANK  WHEN  ZERO.
    05  FILLER                 PIC  X(7).
    05  A-R-AMOUNT-PAST-DUE    PIC  $$,$$$.99   BLANK  WHEN  ZERO.
```

إذا نقلت القيمة صفر إلى A- R- DAYS- PAST- DUE فإنه يطبع على هيئة bbbb ( بدلا من طباعته على هيئة bb0). وإذا نقل صفر فى A- R- AMOUNT- PAST- DUE فإنه يطبع على هيئة bbbbbbbbb ( بدلا من طباعته على هيئة bbbbb$.00 ) . النتيجة هى طباعة القيم غير الصفرية فقط فى التقرير ، وهذا يركز انتباه المستفيد على العملاء الذين عليهم ديون .

**مثـــال ٥ – ٤١ :**

يمكن أن يظهر التقرير المطبوع بواسطة مثال ٥ – ٤٠ ، بدون استخدام BLANK WHEN ZERO ، على النحو التالى :

| NAME | TYPE PAYMENT | DAYS PAST | AMOUNT PAST |
|---|---|---|---|
| FRANK | A 7 | 0 | $.00 |
| ACE | A 7 | 0 | $.00 |
| CARTER | B 3 | 5 | $25.32 |
| SMITH | A 7 | 0 | $.00 |
| PERELMAN | B 5 | 63 | $587.00 |
| CONVERSE | A 7 | 0 | $.00 |
| GETZ | B 3 | 0 | $.00 |

لاحظ كيف تتطلب البيانات المعنوية انتباها آدميا، يمكن أن يضيع فى زحام الأصفار .

## ٥ – ١٣ جزء التضبيط (شكل ٥ . ٧ – السطر رقم 5)

يمكن استخدام جزء التضبط JUSTIFIED مع العناصر الحرفية عددية (PIC X)، أو العددية (PIC A) فقط . عادة ، وبصورة تقليدية .. تضبط البيانات الحرفية عددية والحرفية من ناحية اليسار left- justifird فى الحقل ، ويتسبب هذا الجزء فى ضبطها من ناحية اليمين right- justified ، ونادراً ما يستخدم هذا الجزء .

**مثـــال ٥ – ٤٢ :**

```
05  FIELD-A     PIC X(5).
05  FIELD-B     PIC X(5)     JUSTIFIED RIGHT.
```

افرض أن القيمة CAT نقلت إلى كل من FIELD-A و FIELD-B . عند ذلك ينتج مايلى :

| *Field* | *Prints As* |
|---|---|
| FIELD-A | CAT*bb* |
| FIELD-B | *bb*CAT |

**مثـــال ٥ – ٤٣ :**

```
05  FIELD-A     PIC X(3).
05  FIELD-B     PIC X(3)     JUSTIFIED RIGHT.
```

افرض أن القيمة SMILES نقلت إلى كل من : FIELD-A و FIELD-B . وعند ذلك ينتج مايلى :

| | *Field* | *Prints As* |
|---|---|---|
| (*a*) | FIELD-A | SMI |
| (*b*) | FIELD-B | LES |

تعليقات : ( أ ) إذا كانت القيمة أطول من الحقل فتحذف العناصر المعرفة باستخدام PIC A , PIC X من نفس الناحية التى يحدث فيها ملء فراغات إذا كان طول القيمة أقصر من الحقل . وحيث أن التضبيط التقليدى يكون من ناحية اليسار يحدث الحذف من ناحية اليمين وتنقل الثلاثة رموز الموجودة فى أقصى اليسار إلى الحقل .

(ب) يتسبب جزء JUSTIFIED فى الضبط من ناحية اليمين . وعلى هذا تنقل الثلاثة الرموز الموجودة فى أقصى اليمين من القيمة إلى الحقل المعرف بواسطة PICX (3) .

## ٥ - ١٤ جـــزء الحـــدوث (شكل ٥ . ٧ - السطر رقم 6)

يسمح جزء الحدوث OCCURS للمبرمج بإنتاج هيكل بيانات ، عادة مايسمى بجدول table ، أو منظومة array ، ونناقش معالجة الجداول فى الفصل العاشر .

**مثـال ٥ - ٤٤ :**

```
01  ITEM-DESCRIPTION-RECORD.
    05  ITEM-NUMBER               PIC X(7).
    05  ITEM-WAREHOUSE-BIN        PIC X(2).
    05  ITEM-ON-HAND              PIC S9(5)              COMP-3.
    05  ITEM-SUPPLIERS                        OCCURS     10 TIMES.
        10  ITEM-SUPPLIER-NAME    PIC X(20).
        10  ITEM-SUPPLIER-ID      PIC X(5).
        10  ITEM-SUPPLIER-COST    PIC S9(5)V99           COMP-3.
    05  ITEM-PRICE                PIC S9(5)V99           COMP-3.
```

تتكرر مجموعة العناصر ITEM- SUPPLIERS عدد 10 مرات فى السجل ، مكونة بذلك جدولا table . ويجب أن يستخدم المبرمج دليلا subscript أو فهرسا index فى جزء الإجراءات يحدد محتوى الجدول الذى يجرى تشغيله عليه .

```
MOVE 'NUTS AND BOLTS, INC.' TO ITEM-SUPPLIER-NAME (WHICH-SUPPLIER)
```

للعنصر العددى WHICH- SUPPLIER قيمة تقع بين 1 ، 10 ويحدد العناصر العشرة التى يجرى عليها تشغيل . يجب أن تعرف مثل هذه الأدلة باستخدام USAGE COMP .

```
05  WHICH-SUPPLIER      PIC S9(2) ' COMP.
```

## ٥ - ١٥ جــزء الإشـــارة

يمكن استخدام جزء الإشارة SIGN للعناصر العددية المستخدم فى تعريفها S ، ولـهـا USAGE DISPLAY فقط، والبدائل التالية ممكنة .

- SIGN IS TRAILING

إشارة العملية هى جزء من التمثيل الداخلى للرقم الموجود على أقصى اليمين ولا تتطلب أى موقع تخزين إضافى . وهذا هو الوضع التقليدى لكوبل IBM OS/VS .

- SIGN IS LEADING

إشارة العملية هى جزء من التمثيل الداخلى للرقم الموجود على أقصى اليسار . وهو لايتطلب أى موقع تخزين إضافى .

- SIGN IS TRAILING SEPARATE CHARACTER

تضاف إشارة السالب أو الموجب فى التمثيل الداخلى للعدد ، وذلك باستخدام شفرة EBCDIC أو شفرة ASCII . فى هذه الحالة لاتمثل S الموجودة فى PIC بايت إضافيًا من التخزين للعنصر. وإذا لم توجد الاشارة فى القيمة الموضوعة في عنصر البيانات هذا اثناء تنفيذ البرنامج ينتهى البرنامج نهاية غير طبيعية .

- SIGN IS LEADING SEPARATE CHARACTER

يسبق تمثيل شفرة EBCDIC أو شفرة ASCII لاشارة السالب او الموجب، التمثيل الداخلى للعدد. ويمثل رمز الصورة S مرة أخرى بايت إضافيًا لموقع تخزين مطلوب ، وتتناظر الإشارة الرائدة المنفصلة LEADING SEPARATE الطريقة التى يكتب بها الإنسان الأعداد السالبة . فإذا لم توجد الإشارة فى القيمة المنقولة إلى هذا العنصر أثناء التنفيذ ، فينتهى البرنامج نهاية غير طبيعية .

**مثـــال ٥ – ٤٥ :** DATA-ITEM PIC S9(3)

بالقيمة 123+ ، يبين شكل (٥ – ٨) التمثيل الداخلى لاسم البيانات DATA-ITEM ، لعديد من أجزاء SIGN .

| | | |
|---|---|---|
| SIGN IS TRAILING: | 1 \| 2 \| 3+ | (3 bytes) |
| SIGN IS LEADING: | +1 \| 2 \| 3 | (3 bytes) |
| SIGN IS TRAILING SEPARATE CHARACTER: | 1 \| 2 \| 3 \| + | (4 bytes) |
| SIGN IS LEADING SEPARATE CHARACTER: | + \| 1 \| 2 \| 3 | (4 bytes) |

شكل ( ٥ – ٨ )

ينتج عن البديلين LEADING , SEPARATE CHARACTER برنامج هدف أقل كفاءة بالنسبة لأجهزة كمبيوتر IBM 370 . من الأفضل بصفة عامة أن تحذف SIGN ويقبل التنفيذ التقليدى لإشارة العملية على أى نظام كمبيوتر .

## ٥ - ١٦ جــزء التوافـــق (شكل ٥ . ٧ - السطر رقم 8)

يمكن، على بعض أجهزة مثل IBM 370 ، معالجة العناصر العددية المعرفة باستخدام USAGE COMP بكفاءة أكثر إذا ما ضبطت aligned مواقع الذاكرة المحددة للعنصر على الحد boundary المناسب . إذ يمكن تشغيل عنصر معرف باستخدام COMP يشغل 2 بايت بكفاءة أعلى إذا كان عنوان أول بايت يقبل القسمة على 2 ، والحقل المعرف باستخدام COMP ويشغل 4 بايت، بكفاءة أعلى إذا مابدء ببايت عنوان يقبل القسمة على 4 .

جزء التوافق (SYNCHRONIZED (SYNC يتسبب فى أن يقوم المترجم بتحديد موقع العناصر المعرفة باستخدام COMP بحيث أنها تكون مضبوطة على الحد الأكثر كفاءة . قد يتطلب هذا إدخال بعض البايت الراكدة slacle bytes بين الحقل السابق فى السجل والحقل الذى يضبط ، وتقدم هذه تلقائيا بواسطة المترجم .

**مثـــال ٥ - ٤٦ :**

بمعرفة وصف السجل التالى :

| *Length in Bytes* | | | | |
|---|---|---|---|---|
| | 01 | ORDER-RECORD. | | |
| 4 | 05 | ORDER-NUMBER | PIC X(4). | |
| 5 | 05 | ORDER-CUSTOMER-ID | PIC X(5). | |
| 6 | 05 | ORDER-ITEM-NO | PIC X(6). | |
| 2 | 05 | ORDER-QUANTITY | PIC S9(4) | COMP SYNC. |
| 4 | 05 | ORDER-UNIT-PRICE | PIC S9(5)V99 | COMP SYNC. |

يمكن أن يكون تخطيط الذاكرة للسجل ORDER- RECORD كما يلى ( للبايت الموجود فى اقصى اليسار .. العنوان 0).

| 4 bytes | 5 bytes | 6 bytes | 1 byte | 2 bytes | 2 bytes | 4 bytes |
|---|---|---|---|---|---|---|
| Order # | Customer ID | Item # | *slack* | Quantity | *slack* | Unit Price |

نظرا لأن البايت الراكدة هى مخازن مفقودة ، ويمكن أن تقود إلى أخطاء فى تحديد أطوال السجلات ، يوصى باستخدام COMP SYNC كدلائل فقط على أجهزة كمبيوتر IBM-370 .

**مثـال ٥ – ٤٧ :**

من الأفضل تعريف WHICH- SUPPLIER فى مثال ٥ – ٤٤، كما يلى :

```
05  WHICH-SUPPLIER      PIC S9(2)  COMP SYNC.
```

يجب ملاحظة انه فى كوبل IBM OS/VS تهمل RIGHT, LEFT الموجودة فى SYNCHRONIZED .

## ٥ – ١٧ جزء إعادة التعريف (شكل ٥ . ٧ – السطر رقم 1)

يستخدم جزء إعادة التعريف REDEFINES فى تقديم وصف بديل لعنصر بيانات فردى أو مجموعة عناصر . وعند استخدام REDEFI NES يجب أن يكون أول جزء يتبع رقم المستوى واسم البيانات .

```
level-number   data-name-1      REDEFINES data-name-2
```

Data- name-1 هو الاسم البديل لمنطقة بيانات ، بينما Data- name-2 هو اسمه السابق . ويجب أن يكون رقم المستوى المحدد لـ Data- name-1 مثل رقم مستوى Data- name-2 تماما .

**مثـال ٥ – ٤٨ :**

```
FD  CUSTOMER-MASTER-FILE
    RECORD CONTAINS 105 CHARACTERS
    LABEL RECORDS ARE STANDARD

01  CUSTOMER-MASTER-RECORD.

    05  CUSTOMER-ID                          PIC X(6).
    05  CUSTOMER-NAME                        PIC X(30).
    05  CUSTOMER-ADDRESS                     PIC X(30).
    05  CUSTOMER-REGULAR-DATA.
        10  CUSTOMER-BILLING-TYPE            PIC XX.
        10  CUSTOMER-CREDIT-LIMIT            PIC S9(5)V99   COMP-3.
        10  CUSTOMER-WEIGHT-LIMIT            PIC S9(3)V9    COMP-3.
    05  CUSTOMER-PREFERRED-DATA
        REDEFINES CUSTOMER-REGULAR-DATA.
        10  PREFERRED-DISCOUNT-RATE          PIC SV99       COMP-3.
        10  PREFERRED-LOCATION-CODE          PIC X(3).
        10  PREFERRED-SHIPPING-MODE          PIC X(3).
        10  PREFERRED-BILLING-TYPE           PIC X.
    05  CUSTOMER-CITY-STATE-ZIP.
        10  CUSTOMER-CITY                    PIC X(19).
        10  CUSTOMER-STATE                   PIC X(2).
        10  CUSTOMER-ZIP                     PIC X(9).
```

حيث إن الملف محفوظ على قرص ، فيمكن استخدام COMP-3 للعناصر العددية التى يجرى عليها حسابات . ولايوفر ذلك من المكان على القرص فقط بل تنتج عنه شفرة هدف أكثر كفاءة كذلك . يتسبب جزء REDEFINES فى أن يشغل كل من CUSTOMER- REGULAR- DATA و CUSTOMER- PREFERRED- DATA نفس مواقع الذاكرة تماما ( حيث إن هذا هو FD ) فتكون المواقع جزءا من السجل المنطقى النشط حاليا فى الذاكرة الاحتياطية للملف ) . وعلى هذا توجد منطقة من 9 بايت من السجل المنطقى النشط حاليا ، يمكن أن يكون لها تفسيران مختلفان تماما ( شكل ٥ - ٩ ) .

ويرجع إلى المبرمج ما إذا كان يريد العمل باسماء بيانات تتفق اتفاقاً صحيحا مع محتويات الذاكرة فى أى نقطة من نقاط تنفيذ البرنامج . مثال ذلك .. يمكن أن يكون لكل العملاء المفضلين CUSTOMER- ID ينتهى بصفر. بعد قراءة CUSTOM- ER- MASTER- RECORD .. يمكن أن يختبر البرنامج آخر رمز من CUSTOMER- ID . إذا لم يكن صفرا فسوف يجرى تشغيل لعناصر CUSTOMER- REGULAR- DATA أما إذا كان صفرا .. فسوف يجرى تشغيل CUSTOM- ER- PREFERRED- DATA

CUSTOMER-PREFERRED-DATA

| Discount Rate Pic SV99 COMP-3 (2 bytes) | Location Code Pic X(3) (3 bytes) | Shipping Mode Pic X(3) (3 bytes) | Billing Type Pic X (1 byte) |
|---|---|---|---|

. . . part of currently active logical record in buffer . . .

| Billing Type PIC XX (2 bytes) | Credit Limit PIC S9(5)V99 COMP-3 (4 bytes) | Weight Limit PIC S9(3)V9 COMP-3 (3 bytes) |
|---|---|---|

CUSTOMER-REGULAR-DATA

شكل ( ٥ - ٩ )

لاحظ وجود الحرية الكاملة فى إعادة تعريف REDFINES منطقة، بالنسبة الى USAGE, PICTURE وغيرها . القيد الوحيد هو أن طول العنصر المعاد تعريفه يجب أن يتساوى مع طول العنصر الذى يعيد تعريفه . لاحظ كذلك REDFINES يجب أن تؤخذ فى الحسبان عند تحديد طول السجل ، فطول CUSTOMER- MASTER- DATA ، هو 105 بايت وليس 104 .

**مثـــال ٥ - ٤٩ :**

فى بر نامج مثال ٥ . ١٦ العبارة التالية :

```
01  MAILING-LABEL-CITY-STATE-REC
    REDEFINES MAILING-LABEL-NAME-STREET-REC.
```

يمكن أن تتسبب فى خطأ تكوينى ( ويمكن أن تكون غير ضرورية ، حيث يعيد العنصران على المستوى 01 بعضهما البعض ضمنيا implicitly redefine ) . يجب ألا يعاد تعريف العناصر فى مستوى 01 فى قسم الملفات صراحة . ولكن يمكن أن يعاد تعريفها فى مخزن العمل .

**مثال ٥ - ٥٠ :**

عادة يكون اختبار حقول المدخلات مساعداً فى التصحيح ( التأكد من مهمة validation المدخلات ) وذلك بجعله قادرا على الإشارة إلى حقل عددى معين بأن له PIC9, PICX . ويسمح REDEFINE للمبرمج بالحصول عليها بالطريقتين :

```
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
05  NUMERIC-INPUT-FIELD                          PIC  S9(3)V99.
05  NUMERIC-INPUT-FIELD-AS-X
        REDEFINES NUMERIC-INPUT-FIELD            PIC  X(5).
```

## ٥ - ١٨ تحديد طول عنصر البيانات

كمية الذاكرة اللازمة لتخزين عنصر معين تتحدد بواسطة PICTURE , USAGE ؛ حيث يعتبر طول مجموعة العناصر هو مجموع أطوال عناصرها الفردية ، فيكفى معرفة كيفية إيجاد طول أى عنصر فردى .

- DISPLAY USAGE : كل رمز صورة ( باستثناء V, S , P ) يتطلب بايت واحداً من التخزين .
- COMP USAGE : حيث إنه لايمكن تنقيح عناصر COMP .. فـان P , 9 , V , S هى رموز الصورة الوحيدة المسموح بها . وكما هو الحال مع عناصر DISPLAY فلا تسهم P , V , S فى كمية الذاكرة اللازمة . ولسوء الحظ ، فإن عدد الرموز فى COMP PICTURE لايحدد بنفسه الطول . وبالنسبة لأجهزة كمبيوتر IBM 370 فإن :

| *Number of "9"s in PIC* | *Length of Item in Bytes* |
|---|---|
| 1 to 4 | 2 |
| 5 to 9 | 4 |
| 10 to 18 | 8 |

- COMP-3 USAGE : الرموز P , 9 , V , S هى رموز الصورة الوحيدة المسموح بها . وكما هو الحال مع COMP .. فإن طول عناصر COMP-3 يعتمد على عدد الأرقام فى العدد ( عدد 9 فى الصورة ). إذا كان هناك K من 9 (حيث $1 \leq k \leq 18$) فى صورة COMP-3 .. فإن عدد البايت فى العنصر يكون 1+ [ K/2 ] .

حيث [ K/2 ] تعنى أكبر رقم أقل من أو يساوى K/2 ( أى حذف أى كسر ) .

مثـال ٥ - ٥١ :

```
01  DISK-INVENTORY-RECORD.
    05  INVENTORY-PART-NUMBER          PIC X(8).
    05  INVENTORY-DESCRIPTION          PIC X(30).
    05  INVENTORY-ON-HAND              PIC S9(5)      COMP.
    05  INVENTORY-ON-ORDER             PIC S9(4)      COMP.
    05  INVENTORY-DISCOUNT             PIC S9(3)V99   COMP.
    05  INVENTORY-PRICE-WE-CHARGE      PIC S9(5)V99   COMP.
    05  INVENTORY-ORDERED-DATE.
        10  INVENTORY-ORDERED-MONTH    PIC XX.
        10  INVENTORY-ORDERED-DAY      PIC XX.
        10  INVENTORY-ORDERED-YEAR     PIC XX.
    05  INVENTORY-PERCENT-SCRAPPED     PIC SVP9       COMP.
```

يجب أن يكون لدينا مايلى، على أجهزة IBM 370 :

| *Data Name* | *Length in Bytes* | *Comment* |
|---|---|---|
| DISK-INVENTORY-RECORD | 60 | record length is sum of field lengths |
| INVENTORY-PART-NUMBER | 8 | one byte for each PIC X |
| INVENTORY-DESCRIPTION | 30 | one byte for each PIC X |
| INVENTORY-ON-HAND | 4 | 5 to 9 digits COMP: 4 bytes |
| INVENTORY-ON-ORDER | 2 | 1 to 4 digits COMP: 2 bytes |
| INVENTORY-DISCOUNT | 4 | 5 to 9 digits COMP: 4 bytes |
| INVENTORY-PRICE-WE-CHARGE | 4 | 5 to 9 digits COMP: 4 bytes |
| INVENTORY-ORDERED-DATE | 6 | field length is sum of subfield lengths |
| INVENTORY-ORDERED-MONTH | 2 | one byte for each PIC X |
| INVENTORY-ORDERED-DAY | 2 | one byte for each PIC X |
| INVENTORY-ORDERED-YEAR | 2 | one byte for each PIC X |
| INVENTORY-PERCENT-SCRAPPED | 2 | 1 to 4 digits COMP: 2 bytes (note only one 9 in PIC) |

مع استخدام COMP-3 ، بدلا من COMP ، تحدث التغييرات التالية فى أطوال الحقول السابقة .

| *Data Name* | *Length in Bytes* | *Comment* |
|---|---|---|
| DISK-INVENTORY-RECORD | 58 | 2 bytes saved by using COMP-3 |
| INVENTORY-ON-HAND | [5/2] + 1 = 3 | saves 1 byte versus COMP; there are five "9"s in PICTURE |
| INVENTORY-ON-ORDER | [4/2] + 1 = 3 | takes 1 extra byte versus COMP; there are four "9"s in PICTURE |
| INVENTORY-DISCOUNT | [5/2] + 1 = 3 | saves 1 byte versus COMP |
| INVENTORY-PRICE-WE-CHARGE | [7/2] + 1 = 4 | same as COMP |
| INVENTORY-PERCENT-SCRAPPED | [1/2] + 1 = 1 | saves 1 byte versus COMP |

## ٥ - ١٩ قسم مخزن العمل

يسمح قسم مخزن العمل من جزء البيانات للمبرمج بتعريف مخازن رئيسية لمناطق بيانات ، يحتاجها البرنامج ولاتكون جزءا من أى سجل من السجلات المنطقية التى يجرى عليها البرنامج تشغيلا ( انظر المشكلة ٢١ - الفصل الثانى ) . ويمكن وصف بيانات قسم مخزن العمل بطريقة تشبه تماما وضعها فى قسم الملفات ، مع الاختلافات التالية :

( ١ ) حيث ان حقول مخزن العمل ليست جزءا من سجلات منطقية .. فلا توجد محتويات FD فى قسم مخزن العمل .

( ٢ ) هناك رقم مستوى جديد 77، يستخدم فى تعريف حقول مخزن العمل التى لاتكون جزءا من هيكل سجل .

( ٣ ) هناك جزء جديد لوصف البيانات (VALUE) ، يحدد القيمة التى يجب أن يأخذها عنصر قسم مخزن العمل، عند بدء تنفيذ البرنامج .

## المستوى 77

بينما تكون كل الحقول فى قسم الملفات إما سجلات منطقية أو أجزاء من سجلات منطقية ، فالعديد من حقول قسم مخزن العمل عبارة عن عناصر مستقلة independent items ، ليست لها علاقة باى حقول اخرى فى مخزن العمل . ويمكن أن يعطى لكل عنصر من العناصر المستقلة رقم المستوى 77 مع وضعها فى بداية مخزن العمل . إلا أننا نوصى ، كلما كان ذلك ممكنا ، أن تجمع العناصر المستقلة فى سجلات مخزن عمل ، لها المستوى 01 ( كما فى حالة قسم الملفات ) . وهذا يسهل من تصحيح البرنامج وصيانته .

**مثال ٥ - ٥٢ :**

قارن الصيغتين التاليتين من قسم مخزن العمل :

```
●    77  NUMBER-OF-CHECKS-PRINTED        PIC S9(5)     COMP-3.
     77  NUMBER-OF-OVERTIME-WORKERS      PIC S9(5)     COMP-3.
     77  TAX-TABLE-SUBSCRIPT             PIC S9(4)     COMP SYNC.
     77  ALL-RECORDS-PROCESSED-SW        PIC X.
     77  DEDUCTION-TABLE-SUBSCRIPT       PIC S9(4)     COMP SYNC.
     77  BELONGS-CREDIT-UNION-SW         PIC X.

     01  PAYROLL-REPORT-LINE-AREA.
         05  PAYROLL-EMPLOYEE-ID         PIC X(5).
         05  FILLER                      PIC X(3)      VALUE SPACES.
         05  PAYROLL-EMPLOYEE-NAME       PIC X(20).
         05  FILLER                      PIC X(4)      VALUE SPACES.
         05  PAYROLL-GROSS-PAY           PIC $$$,$$$.99.

●    01  WORKING-COUNTERS.
         05  WORKING-NUMBER-CHECKS-PRINTED    PIC S9(5)     COMP-3.
         05  WORKING-NUM-OVERTIME-WORKERS     PIC S9(5)     COMP-3.

     01  WORKING-PROGRAM-SWITCHES.
         05  ALL-RECORDS-PROCESSED-SW         PIC X.
         05  BELONGS-CREDIT-UNION-SW          PIC X.
```

```
01  WORKING-TABLE-SUBSCRIPTS.
    05  TAX-TABLE-SUBSCRIPT              PIC S9(4)         COMP SYNC.
    05  DEDUCTION-TABLE-SUBSCRIPT        PIC S9(4)         COMP SYNC.

01  PAYROLL-REPORT-LINE-AREA.
    05  PAYROLL-EMPLOYEE-ID              PIC X(5).
    05  FILLER                           PIC X(3)          VALUE SPACES.
    05  PAYROLL-EMPLOYEE-NAME            PIC X(20).
    05  FILLER                           PIC X(4)          VALUE SPACES.
    05  PAYROLL-GROSS-PAY                PIC $$$,$$$.99.
```

فى الصيغة الثانية ، تم تجميع العناصر الستة الاصلية على مستوى 77 فى ثلاثة أجزاء : عدادات ومفاتيح برنامج ودلائل. تعالج كل مجموعة عناصر كمجموعة حقول على المستوى 01 . اسم البيانات على المستوى 01 يكون وصفيا لفئة عناصر البيانات المعرفة . وهذه الصيغة الثانية أفضل كثيرا فى تنظيمها عن الصيغة الأولى .

**مثـال ٥ - ٥٣ :**

بالرغم من أن جزء PIC لايسرى على مجموعة عناصر.. فيمكن تحديد SYNCHRONIZED, USAGE على مستوى المجموعة ، وفى هذه الحالة.. تسرى على كل العناصر الموجودة فى المجموعة . ونشعر بأن هذه العملية يجب أن تستخدم بشكل محدود، إذا استخدمت على الاطلاق ، حيث إنها تزيد من احتمال عدم تمييز مبرمج الصيانة USAGE الصحيحة للعنصر. الموقع الآمن نسبيا لاستخدام USAGE لمجموعة فى تجميعات العناصر المستقلة ( عناصر المستوى 77 ) فى قسم مخزن العمل ( كما فى مثال ٥ - ٥٢ ، WORKING- COUNTERS COMP-3 ) .

## جـــــزء القيمـــــة

عندما يستخدم جزء

$$\left[\underline{\text{VALUE}}\ \text{IS}\ \left\{\begin{matrix}\text{literal}\\ \text{figurative constant}\end{matrix}\right\}\right]$$

فإنه يحدد قيمة عنصر بيانات مخزن العمل عند بدء تنفيذ البرنامج . إذا لم يستخدم جزء VALUE .. فيكون لعناصر البيانات محتويات غير متوقعة فى البداية ، ويقول المبرمجون إن مثل هذه العناصر تحتوى على نفايا أو إنها غير معرفة . لاتهم القيم الابتدائية لعديد من العناصر نظرا لأن منطق البرنامج يدخل بيانات ذات معنى فيها ( حاذفا النفايا ) قبل محاولة استخدام محتويات الموقع .

هناك نوعان من الثوابت، وستة أنواع من الثوابت الاستعارية .

• **الثـوابت غيـر العدديـة** Nonnumeric literals : يحتوى الثابت غير العددى على أى سلسلة من الرموز موضوعة بين علامتى تنصيص ( تسمى quoted string - القسم الثانى - الفصل الثانى ) . إذا كانت هناك رغبة فى تمثيل علامة تنصيص فى ثابت غير عددى.. تستخدم علامتى تنصيص متتاليتين ، ولاتعتبر إلا علامة تنصيص واحدة كجزء من الثابت .

**مثـال ٥ – ٥٤ :**

```
05   CUSTOMER-NAME   PIC X(15)   VALUE """"KWIK"" CLEAN".
```

يحدد القيمة "KWIK" CLEAN ( يمثل كل زوج من علامات التنصيص المزدوجة داخل الثابت علامة تنصيص مزدوجة واحدة فى القيمة ) .

تستخدم بعض المترجمات علامة تنصيص فردية ( أو فاصلة ) ، بدلا من علامة التنصيص المزدوجة . ويحدد كوبل 1974 النمطى استخدام علامات تنصيص مزدوجة ، إلا إنه يجب على المبرمج أن يرجع – بالطبع – إلى نظام الكمبيوتر المتاح له استخدامه .

```
05   CUSTOMER-NAME   PIC X(15)   VALUE 'PERELMAN' 'S PETS'.
```

• **الثوابت العددية** Numeric literals : تتكون من الأرقام من صفر إلى ٩ وإشارة الموجب وإشارة السالب والعلامة العشرية . تعتمد أقصى قيمة للثابت العددى على نظام الكمبيوتر وهى ١٠^١٨ – ١ فى كوبل IBM OS/VS ( أى ١٨ خانة ) .

فإذا لم تحدد الإشارة .. تفترض الثوابت العددية أنها موجبة . وإذا لم توجد علامة عشرية فى ثابت عددى ، فيقال إنه من النوع الصحيح integer .

• **الثوابت الاستعارية** Figurative constamts : الثابت الاستعارى هو كلمة كوبل محجوزة، تمثل قيمة ثابتة يمكن استخدامها فى أى مكان فى البرنامج فى موقع الثابت المناظر لها . ويعرض جدول ٥ – ٢ الثوابت الاستعارية الموجودة فى كوبل IBM SO/VS

| القيمة | نوع الثابت الذى تمثله | كلمة كوبل المحجوزة |
|---|---|---|
| صفر ( يتكرر عدد مناسباً من المرات ، يضبط المترجم الصفر على إنه COMP ، أو DISPLAY ) | عددى | ZERO<br>ZEROS (or ZEROES) |
| فراغ ( يتكرر عدد مناسب من المرات ) | غير عددى | SPACE SPACES |
| أعلى رمز في تسلسل التتابع ( يتكرر عدداً مناسباً من المرات ) . | غير عددى | HIGH-VALUE<br>HIGH- VALUES |
| أقل رمز فى تسلسل التتابع ( يتكرر عدد مناسب من المرات ) . | غير عددى | LOW- VALUE<br>LOW- VALUES |
| علامة تنصيص ( فردية أو مزدوجة ) وتتكرر عدداً مناسب من المرات . | غير عددى | QUOTE<br>QUOTES |
| يمكن أن يكون literal أى سلسلة رموز بين علامتى تنصيص ( تتكرر السلسلة ) . | غير عددى | ALL literal |

جدول ٥ - ٢

**مثـال ٥ - ٥٥ :**

لاحظ استخدام الثوابت والثوابت الاستعارية فيمايلى :

```
. . . . . . . . . . . . . . . . . . . . . . .
WORKING-STORAGE SECTION.
01  ALL-WORKING-STORAGE.
    05  FILLER                  PIC X(30)
                                VALUE "WORKING-STORAGE BEGINS HERE".
    05  LAST-RECORD-FLAG        PIC X(3)
                                VALUE HIGH-VALUES.
    05  UNDERLINE-AREA          PIC X(100)    VALUE ALL "-".
    05  BLANK-LINE-AREA         PIC X(100)    VALUE SPACES.
    05  INVALID-RECORD-COUNT    PIC S9(4)     COMP-3     VALUE ZERO.
. . . . . . . . . . . . . . . . . . . . . . .
PROCEDURE DIVISION.
    MOVE SPACES TO OUTPUT-MESSAGE-AREA
    MOVE "INVALID PART NUMBER" TO OUTPUT-MESSAGE
    MOVE 200.95 TO DAILY-QUOTA
    MOVE ALL "*" TO PAID-OUT-MESSAGE-AREA
    MOVE LOW-VALUES TO EMPLOYEE-MASTER-ID
```

لاحظ بصفة خاصة استخدام الثابت غير العددى فى تحديد بداية مخزن العمل ، وهذا يسهل من إيجاد مخزن العمل فى الزحام dump .

## القيمــة والنقـــل

كما سبق أن رأينا فإن جزء VALUE يمكن استخدامه فى وضع قيم ابتدائية فى عناصر مخزن العمل .

```
05   INVALID-RECORD-COUNT   PIC S9(4)   COMP-3   VALUE ZERO.
```

ويمكن الان استخدام عبارة MOVE من جزء الإجراءات فى بداية تنفيذ البرنامج، لوضع قيم ابتدائية للعناصر كذلك .

```
MOVE ZERO TO INVALID-RECORD-COUNT
```

ونقدم المقترحات التالية الخاصة بهاتين الطريقتين :

( ١ ) إذا احتفظ العنصر الذى تحدد له قيمة ابتدائية بقيمته الابتدائية أثناء تنفيذ البرنامج .. استخدم جزء VALUES .

( ٢ ) إذا تغيرت قيمة عنصر البيانات أثناء تنفيذ البرنامج، ويتطلب منطق البرنامج أن يكون للعنصر قيمة ابتدائية محددة.. استخدم عبارة MOVE .

( ٣ ) إذا لم يكن العنصر فى مخزن العمل ( باستثناء العناصر الموجودة على المستوى 88 ، انظر القسم الثامن – من الفصل السابع ) فلا يمكن استخدام جزء VALUE .

( ٤ ) إذا كانت القيمة الابتدائية لعنصر البيانات غير مناسبة للبرنامج .. فلا تضع قيمة ابتدائية للعنصر .

## ٥ – ٢٠ نمطيات كتابة الشفرة

نعيد هنا صياغة نمطيات كتابة الشفرة لجزء التعريف، وجزء الأوساط وجزء البيانات .

( ١ ) مقاطع جزء التعريف مشمولة فى كوبل ANS النمطى؛ لان المعلومات التى تمثلها مهمة (لاتهمل محتويات جزء التعريف).

( ٢ ) هناك مرونة معتبرة فى كتابة جزء التعريف ؛ بما يجعله سهل القراءة .

( ٣ ) استخدم أسطر تعليقات فى نهاية جزء التعريف لتلخيص مايؤديه البرنامج .

( ٤ ) استخدم نقاط مرتبة عندما تكون هناك عدة أجزاء ( كما هو الحال فى مقطع الأسماء الخاصة وفى وصف الملف FD ) .

( ٥ ) استخدم أرقام مستويات متزايدة بأكثر من 01 ، سامحا بإدخال أرقام بينها مستقبلا . ( يستخدم فى هذا الملخص (01، 05 ، 10 ، 15 ، 20 ، ... ) .

( ٦ ) قم بعمل ترحيل لتوضيح هيكل أرقام المستويات .

( ٧ ) يجب أن تكون لكل اسم بيانات سابقة prefix ( أو لاحقة suffix ) تصاحبه بالسجل الذى يتبعه العنصر .

( ٨ ) يجب أن تكون بقية اسم البيانات وصفية بقدر الأمكان ( تذكر أنك تستطيع استخدام 30 رمز فى الاسم الواحد ).

( ٩ ) اكتب عبارات SELECT مرتبة فى احدى الصيغ النمطية ( مثل ملفات المدخلات ثم ملفات المدخلات والمخرجات ، ثم ملفات المخرجات ، واستخدام نفس الترتيب فى FD ).

(١٠) لاتستخدم عناصر على المستوى 77 فى مخزن العمل ، وأجمع بدلا من ذلك العناصر فى فئات معنوية ، تحت اسم وصفى على المستوى 01 .

(١١) استخدم ترتيبا نمطيا معينا فى كتابة عناصر مخزن العمل ليشتمل على : مفاتيح البرنامج، والإشارات، والعدادات، والثوابت الاستعارية، والثوابت، ومناطق الإجماليات ( المركمات ) والدلائل والجداول، ونسخ العمل للسجلات المنطقية للملفات . ويجب أن ترتب هذه الأخيرة مثل ترتيب ظهورها فى FD ، عرف لملفات الطباعة نسخ العمل لأسطر العناوين أولا، يليها أسطر الجسم وأخيرا أسطر الإجماليات والنهاية.

(١٢) كلما كان ممكنا اضبط VALUE , SYNC , USAGE , PIC فى الأعمدة ، بحيث يبدأ كل منها من عمود معين .

(١٣) إذا كان ممكنا .. حدد وظيفة عنصر البيانات فى اسم البيانات ، مثل جعل كل مفاتيح البرنامج تنتهى بكلمة SWITCH أو باختصارها SW، وكل الإشارات تنتهى بكلمة FLAG، ومناطق الإجماليات تنتهى بكلمة TOTAL، وتنتهى كل العدادات بكلمة COUNTER أو COUNT .

(١٤) تقدم بعض المترجمات توسعات فى كوبل ANS ، تسمح للمبرمج بفصل العناصر فى قائمة برنامج المصدر، لتسهيل القراءة اكثر . فمثلا.. يتسبب كوبل IBM OS/VS باستخدام EJECT المكتوبة فى أى مكان من السطر في المنطقة B ، فى بدء السطر التالى من القائمة في قمة صفحة جديدة . وباستخدام SKIP1 او SKIP2 او SKIP3 المكتوبة فى أى مكان فى المنطقة B من السطر ، يترك المترجم العمود المحدد له من الأسطر دون طباعتها فى القائمة، ويمكن استخدام هذه السمات؛ للتأكد أن كل جزء، وكل قسم يبدأ فى صفحة جديدة من القائمة ، وأن المقاطع مفصولة عن بعضها البعض ( مع وجود أسطر فارغة قبل اسم المقطع ) . يمكن تشغيل SKIP 1 ، و SKIP 2 ، و SKIP 3 أسرع بواسطة المترجم ، عما فى حالة استخدام أسطر فارغة موجودة داخل البرنامج لعمل الفراغات .

## اسـئلة مراجعـة

٥ - ١ ما الغرض من جزء البيانات ؟ ومن قسم الملفات ومن قسم مخزن العمل ؟

٥ - ٢ ما نوع عناصر البيانات المتوقع وجوده فى مخزن العمل ؟

٥ - ٣ ما الغرض من الأجزاء التى تكتب على المستوى 01 فى جزء البيانات ؟

٥ - ٤ وضح العلاقة بين FD وعبارات SELECT ؟

٥ - ٥ ما المعلومات التى تحدد فى FD ؟

٥ - ٦ وضح كيف يوفر التجميع من المواقع على الشريط أو القرص .

٥ - ٧ ميز بين السجلات المنطقية والسجلات الواقعية ؟

٥ - ٨ عرف معامل التجميع .

٥ - ٩ ما المقصود بـ « فراغات مابين السجلات، وفراغات مابين المجموعات » ؟

٥ - ١٠ وضح كيف يسرع التجميع من تشغيل الملفات التتابعية للقرص او الشريط .

٥ - ١١ وضح تأثير التجميع على تشغيل الملفات عشوائيا .

٥ - ١٢ وضح كيف تعالج السجلات متغيرة الطول بواسطة نظم IBM SO/VS .

٥ – ١٣ وضح كيف تحسب قيم BLOCK CONTAINS لسجلات متغيرة الطول .

٥ – ١٤ لماذا يعد جزء RECORD CONTAINS اختياريا ؟ ومتى يكون اجباريا ؟

٥ – ١٥ وضح كيف تحسب قيم RECORD CONTAINS لسجلات متغيرة الطول .

٥ – ١٦ ما المعنى الخاص لجزء BLOCK CONTAINS 0 RECORD فى كوبل IBM SO/VS ؟

٥ – ١٧ وضح استخدام عناوين الملفات لملفات الشرائط والأقراص .

٥ – ١٨ ما حجم جدول المحتويات VTOC أو الدليل ؟

٥ – ١٩ ما الجزء الوحيد اللازم ظهوره فى FD ؟

٥ – ٢٠ أى الملفات يفضل أن تكون لها عناوين ملفات ؟ وأيها يفضل ألا تكون لها عناوين ملفات ؟

٥ – ٢١ لماذا يستخدم DATA RECORD بصفة متكررة ؟ ومتى يمكن أن يكون مفيدا ؟

٥ – ٢٢ اذكر مثالا لملف به أكثر من نوع واحد من السجلات المنطقية . ووضح كيف يمكن استخدام شفرة السجلات record codes فى التمييز بين الأنواع المختلفة .

٥ – ٢٣ عرف العناصر التالية المصاحبة لجزء الخطية LINAGE : (أ) صفحة منطقية (ب) جسم الصفحة (جـ) منطقة نهاية (د) هامش علوى (هـ) هامش سفلى .

٥ – ٢٤ مع أى أنواع الملفات يجب استخدام جزء LINAGE ؟

٥ – ٢٥ ما منطقة المدخلات والمخرجات I/0 area أو الذاكرة الاحتياطية buffer ؟

٥ – ٢٦ وضح العلاقة بين العناصر المعرفة على المستوى 01 فى FD للملف، والذاكرات الاحتياطية للملف .

٥ – ٢٧ ما أرقام المستويات المسموح باستخدامها ؟

٥ – ٢٨ ما الشئ الخاص برقم المستوى 01 ؟

٥ – ٢٩ وضح كيفية استخدام أرقام المستويات فى توضيح هيكل السجل .

٥ – ٣٠ ماهى أهمية الترحيل فى جزء البيانات ؟

٥ – ٣١ عرف : (أ) مجموعة العناصر (ب) العنصر الفردى .

٥ – ٣٢ قدم بعض المقترحات لعمل أسماء لأسماء بيانات .

٥ – ٣٣ عندما يكون للملف أكثر من نوع واحد من السجلات المنطقية ، لماذا يكون كل من النوعين متفقا بصفة عامة فى بايت معين، أو فى مجموعة من البايت ؟

٥ – ٣٤ وضح كيف تسرع الذاكرات الاحتياطية المتعددة من تشغيل الملفات التتابعية .

٥ – ٣٥ فيم يستخدم FILLER ؟

٥ – ٣٦ صف معنى التنقيح editing .

٥ – ٣٧ ما الغرض من جزء PICTURE ؟

٥ – ٣٨ كيف يمكن تكرار رمز صورة بسهولة فى PICTURE ؟

٥ – ٣٩ وضح استخدام رمزى الصورة A و B ؟

٥ - ٤٠ لماذا يتكرر استخدام X دئما كرمز صورة ؟

٥ - ٤١ ناقش مايجب استخدامه فى PICTURE لتمثيل أعداد .

٥ - ٤٢ وضح استخدام S و 9 و V فى تمثيل الأعداد . ميز بين هذه الرموز والرمز X .

٥ - ٤٣ عرف : (أ) الحذف (ب) التقريب .

٥ - ٤٤ لماذا لايتطلب S أو V موقع تخزين فى عنصر البيانات ؟

٥ - ٤٥ متى يجب استخدام S ؟ ولماذا ؟ ومتى يجب حذفها ؟

٥ - ٤٦ ما نتيجة عنصر عددى له رموز S و 9 و V فى صورته ويطبع كما هو فى نظام IBM ؟

٥ - ٤٧ وضح التنقيح العددى باستخدام Z ، والنقطة والفاصلة و $ ، و CR ، و DB ، وإشارة سالب وإشارة موجب .

٥ - ٤٨ وضح حماية الشيكات .

٥ - ٤٩ وضح الفرق بين التنقيح الثابت fixed والمتحرك floated .

٥ - ٥٠ فى أى صورة تطبع الأعداد السالبة عادة فى تقارير الأعمال ؟

٥ - ٥١ وضح استخدام صورة 0 ، و / ، و P .

٥ - ٥٢ حدد USAGE المتاح فى كوبل IBM OS/VS ، وأذكر متى يجب أو لا يجب استخدام كل منها .

٥ - ٥٣ متى يجب أن يعرف العنصر بانه DISPLAY ؟

٥ - ٥٤ ماذا يكون COMP و COMP-3 أكثر كفاءة للحقول فى الحسابات ؟

٥ - ٥٥ اذكر بعض مميزات استخدام COMP او COMP-3 ، للحقول العددية فى سجلات قرص أو شريط ؟

٥ - ٥٦ وضح كيف تحدد طول عنصر بيانات كما يلى (أ) DISPLAY (ب) CONP (جـ) CONP-3 .

٥ - ٥٧ وضح استخدام BLANK WHEN ZERO .

٥ - ٥٨ وضح وظيفة JUSTIFIED .

٥ - ٥٩ ما وظيفة جزء OCCURS ؟

٥ - ٦٠ ناقش البدائل المختلفة لجزء SIGN IS .

٥ - ٦١ وضح ماتفعله SYNCHRONIZED .

٥ - ٦٢ ما البايت الراكدة slack bytes ؟ اربطها بالسؤال عن متى يجب أو لايجب استخدام SYNC ؟

٥ - ٦٣ وضح استخدام REDEFINES مع إعطاء مثال .

٥ - ٦٤ ماالبديل لاستخدام مستوى 77 فى مخزن العمل ؟

٥ - ٦٥ مامعنى نفايا garbage ؟

٥ - ٦٦ اذكر قواعد عمل ثوابت الكوبل، وأذكر الثوابت الاستعارية .

٥ - ٦٧ وضح استخدام جزء VALUE .

٥ - ٦٨ متى يجب أن توضع قيمة ابتدائية لعنصر بيانات باستخدام جزء VALUE ؟ ومتى يجب حدوث ذلك بعبارة MOVE من جزء الإجراءات ؟

## مسائل محلولة

٥ – ٦٩ ما الخطأ فى محتوى قسم مخزن العمل التالى ؟

```
05   MAXIMUM-TOTAL-SALES     PIC S9(5)V99   VALUE HIGH-VALUES.
```

LOW-VALUES و HIGH- VALUES هى ثوابت استعارية غير عددية ، ويمكن استخدامها مع PIC X فقط ولايمكن استخدامها فى الحسابات .

٥ – ٧٠ بمعرفة مايلى :

```
SELECT CUSTOMER-ORDER-FILE    ASSIGN TO ORDERS.
SELECT ORDER-EDIT-REPORT      ASSIGN TO EDITLOG.
SELECT CUSTOMER-MASTER-FILE   ASSIGN TO CUSTMAST.
```

ما أسماء FD التى يجب استخدامها فى قسم الملفات ؟

```
FILE SECTION.
FD  CUSTOMER-ORDER-FILE
. . . . . . . . . . . . . . . . . . . . . . . .
FD  ORDER-EDIT-REPORT
. . . . . . . . . . . . . . . . . . . . . .
FD  CUSTOMER-MASTER-FILE
. . . . . . . . . . . . . . . . . . . . . . . . .
```

٥ – ٧١ افرض أن CUSTOMER- ORDER- FILE فى المسألة السابقة يحتوى على سجلات، طول كل منها 65 بايت ، وبمعامل تجميع 40 لها . وأنها مخزنة حاليا على قرص . اذكر FD الكامل لها .

```
FD  CUSTOMER-ORDER-FILE
    BLOCK CONTAINS 40 RECORDS
    RECORD CONTAINS 65 CHARACTERS
    LABEL RECORDS ARE STANDARD
```

حيث إن الملف موجود على قرص ، .. فإن المطلوب هو عناوين نمطية للملف . فى نظم IBM ، يكون من الأفضل استخدام BLOCK CONTAINS 0 RECORDS ؛ بحيث يمكن أخذ معامل التجميع من عنوان الملف الموجود فعلا .

٥ – ٧٢ إذا كان الملف الموجود فى المسألة السابقة محتويا على 2000 سجل ، ماالعدد اللازم من المدخلات من القرص لتشغيل الملف تتابعيا ؟

مع تجميع 40 سجل فى المجموعة ، يتطلب 2000 سجل عدد 2000/40 أى 50 مجموعة . لأن كل عملية مدخلات من القرص تدخل مجموعة .. فلا تكون هناك حاجة إلا إلى 50 عملية مدخلات فقط .

إذا لم يستخدم التجميع مع الملف .. فلا يوجد إلا سجلاً منطقياً واحداً فى كل سجل واقعى . وعلى هذا .. يجب أن يكون هناك عملية مدخلات لكل سجل منطقى؛ أى 2000 عملية للملف كله .

٥ – ٧٣ لماذا يعتبر عدد عمليات مدخلات أو مخرجات القرص مقياسا مهما لسرعة تشغيل الملف ؟

على عكس الاتصال بالبيانات فى ذاكرة الكمبيوتر ، والذى يمكن أن يحدث فى نانوثانية ( $١٠^{-٩}$ ثانية ) أو أقل ، فأحيانا .. يشمل الاتصال بالبيانات من القرص حركة ذراع اتصال ( الا إذا كانت للقرص رأس ثابتة لكل مسار أو إلا إذا ماحدث أن تواجدت الذراع المتحركة فى المكان المطلوب بالصدفة ) ، ويشمل دائما تأخير دوران بينما تدور المجموعة المطلوبة تحت الرأس . وهذه هى عمليات ميكانيكية ، وهى بطيئة جدا ، بالمقارنة بالعمليات الالكترونية للكمبيوتر : لأقراص IBM من طراز 3308، يكون متوسط وقت الحركة ١٦ ميللى ثانية ومتوسط وقت تأخر الدوران ٨,٣ ميللى ثانية . إذا مااحتوى تشغيل الملف على عدد كبير من الاتصال الواقعى بالقرص ، فمجموع هذه الاوقات يصبح وقتا كبيرا .

٥ – ٧٤ كيف يؤثر التجميع على استغلال المواقع المتاحة على القرص أو الشريط ؟

يوفر التجميع المواقع على القرص او الشريط عن طريق حذف بعض فراغه مابين السجلات . فمثلاً يحتاج الملف الموجود فى المسألة رقم ٧٢ الى ٤ مسارات على قرص من طراز 3308 اذا كان معامل التجميع 40 . اما إذا كان معامل التجميع 1 فانه يحتاج إلى 25 مسارا .

٥ – ٧٥ افرض ان ORDER- EDIT- REPORT فى المسألة رقم ٧٠، سيكون مخرجات على طابع يطبع ١٣٢ رمزا فى السطر الواحد . اكتب FD كاملا لهذا الملف .

```
FD  ORDER-EDIT-REPORT
    RECORD CONTAINS 132 CHARACTERS
    LABEL RECORDS ARE OMITTED
    LINAGE IS 60
       WITH FOOTING AT 55
       LINES AT TOP 3
       LINES AT BOTTOM 3
```

جزء BLOCK CONTAINS محذوف ، لتحديد ان ملف الطباعة غير مجمع . بالنسبة للفات الطابعات يكون هذا الجزء هو LABEL RECORDS ARE OMITTED . يجعل جزء الخطية LINAGE من الاسهل لعبارات WRITEمن جزء الإجراءات أن تتحكم فى موقع الأسطر فى الصفحة . يمكن إلا تستخدم قنوات التحكم فى العربة مع هذا الملف ، حيث إن LINAGE محدد .

٥ – ٧٦ افرض أن CUSTOMER- MASTER- FILE فى المسألة رقم ٧٠ ، له نوعان من السجلات المنطقية : -REG ULAR- CUSTOMER- RECORDS ، وطول كل منها 200 بايت بينما -PREDEFINED- CUSTOMER RECORDS طول كل منها 350 بايت . اكتب FD كاملا ، اذا كان معامل التجميع 40. استخدم BLOCK CONTAINS ... RECORDS .

```
FD  CUSTOMER-MASTER-FILE
    BLOCK CONTAINS 40 RECORDS
    RECORD CONTAINS 200 TO 350 CHARACTERS
    LABEL RECORDS ARE STANDARD
    DATA RECORDS ARE
        REGULAR-CUSTOMER-RECORD
        PREFERRED-CUSTOMER-RECORD

01  REGULAR-CUSTOMER-RECORD . . .
01  PREFERRED-CUSTOMER-RECORD . . .
```

٥ - ٧٧ حل المسألة رقم ٧٦، مستخدما BLOCK CONTAINS... CHARACTERS .

```
FD  CUSTOMER-MASTER-FILE
    BLOCK CONTAINS 8164 TO 14164 CHARACTERS
    RECORD CONTAINS 200 TO 350 CHARACTERS
    LABEL RECORDS ARE STANDARD
    DATA RECORDS ARE
        REGULAR-CUSTOMER-RECORD
        PREFERRED-CUSTOMER-RECORD
```

لايشتمل RECORD CONTAINS على بايت مستخدم فى كلمات واصف السجل والمجموعة ، إلا أنه عند استخدام جزء CHARACTERS فيجب أن يحتوى BLOCK CONTAINS على منطقة من ٤ بايت . عدد البايت المطلوب هو ٨١٦٤ وذلك لشغل مجموعة من الحجم الصغير : وهو ٤٠ سجل منطقى، وطول كل منها طوله ٢٠٠ بايت مضافا إليها ٤٠ كلمة واصف سجل، طول كل منها ٤ بايت ، مضاف اليها كلمة واصف مجموعة من الحجم الكبير هو ١٤١٦٤ بايت .

فى كوبل IBM OS/VS يفضل تحديد BLOCK CONTAINS CHARACTERS بحيث يمكن اخذ حجم المجموعة من عنوان الملف (حيث إن هذا يتواجد فعلا فى الملف الذى له عناوين نمطية) .

٥ - ٧٨ إذا لم يكن الملف موجودا ، ويراد أعداده بواسطة البرنامج ، هل يمكن استخدام ... BLOCK CONTAINS 0 ؟ .

نعم ، إلا أنه فى هذه الحالة.. يجب أن تحدد عبارة لغة تحكم العمل التى تعرف الملف حجما للمجموعة وإلا فإنه سيحدث خطأ .

٥ - ٧٩ صنف وحدات المدخلات والمخرجات التى سبق مناقشتها فى هذا الكتاب ، من ناحية : إذا كانت تدعم أو لا تدعم عناوين ملفات .

| سجلات العناوين محذوفة | سجلات العناوين نمطية |
|---|---|
| ١ - قارئ البطاقات | ١ - قرص له رأس متحركة |
| ٢ - مثقب البطاقات | ٢ - قرص له رأس ثابتة وأسطوانات |
| ٣ - الطابع | ٣ - شريط مغناطيسى ( على بكرة ) |
| | ٤ - شريط مغناطيسى ( فى كاسيت ) |
| | ٥ - قرص مرن |

العناوين مطلوبة لكل أنواع ملفات الاقراص . وتستخدم العناوين – بصفة عامة – لكل أنواع ملفات الشرائط ؛ نظرا للأمن والراحة التى تقدمها .

٥ – ٨٠ اكتب عبارات جزء إجراءات ، تشمل : شفرات سجلات ، وتطبع عناوين بريدية من ملف مثالى رقم ٥ – ١٦ ورقم ٥ – ١٧ .

```
READ MAILING-LABEL-FILE RECORD
IF MAILING-LABEL-NAME-ST-CODE IS EQUAL TO "1"
   MOVE MAILING-LABEL-NAME-ST-NAME        TO LABEL-NAME
   MOVE MAILING-LABEL-NAME-ST-STREET      TO LABEL-STREET
ELSE
   IF MAILING-LABEL-NAME-ST-CODE IS EQUAL TO "2"
      MOVE MAILING-LABEL-CITY-STATE-CITY    TO LABEL-CITY
      MOVE MAILING-LABEL-CITY-STATE-STATE   TO LABEL-STATE
      MOVE MAILING-LABEL-CITY-STATE-ZIP     TO LABEL-ZIP
   ELSE
      DISPLAY "ERROR -- INVALID RECORD CODE:  IGNORED"
```

تدخل عبارة READ سجلا منطقيا، أولا من الملف MAILING- LABEL- FILE . تذكر من مثال ١٧ – الفصل الخامس، أن أول بايت من السجل المنطقى يشغلها شفرة سجل تعرف نوع السجل . افرض أن سجلات الاسم والشارع لها ١ فى أول بايت، وسجلات المدينة، والولاية، والرمز البريدى لها ٢ فى أول خانة . حيث أن أول بايت من كل نوع من أنواع السجلات موصوف بصورة X فلايهم شئ بالنسبة إلى اسم البيانات الذى يستخدمه المبرمج فى الإشارة إليه ( يصل كل من MAILING- LABEL- NAME- ST- CODE و MAILING- LABEL- CITY- STATE- CODE إلى أول بايت من السجل المنطقى ويفسر على أن له الصورة X .)

بعد عبارة READ ، يستخدم البرنامج عبارة IF فى التأكد مما إذا كان أول بايت فى السجل المنطقى به 1 أم لا . فإذا كان فيه 1 ، فينقل الاسم والشارع إلى منطقة العنوان ، إما إذا لم يكن هذا هو الحال، فيتأكد مما إذا كان فيه ٢ أم لا . فإذا لم يكن فيه ١ أو ٢ فيطبع البرنامج رسالة خطأ، ويهمل السجل المنطقى . لاحظ كيفية استخدام أسماء البيانات فى عبارات MOVE وأنها تعتمد تماما على شفرة السجل

٥ – ٨١ يمكن استخدام كل من PIC A و PIC X مع بيانات حرفية . أيهما أفضل ؟

يستخدم عديد من مبرمجى الكوبل PIC X بدلا من PIC A فى وصف البيانات الحرفية . لايوجد عيب فى عمل ذلك ، إلا أنه هناك شئ من المميزات فعلا : فبعض عبارات جزء الإجراءات تسرى على PIC X ، ولاتسرى على PIC A . (انظر الفصلين السادس والسابع) .

٥ – ٨٢ صنف عناصر البيانات التالية صورها على أنها (i) حرفية او (ii) حرفية عددية او (iii) عددية أو (iv) عددية للتنقيح .

| | | | | | |
|---|---|---|---|---|---|
| (*a*) | AABAAA | (*e*) | 99/99/99 | (*i*) | A(10) |
| (*b*) | XXBXXX | (*f*) | $ZZ,ZZZ.99 | (*j*) | 9(3)V99 |
| (*c*) | S9(3)V99 | (*g*) | S99PP | (*k*) | $$$,$$$.99BCR |
| (*d*) | S9(3).99 | (*h*) | X(7) | (*l*) | Z999 |

الحرفية : (a) و (i)

الحرفية عددية : (b) و (h)

العددية : (c) و (g) و (j)

العددية للتنقيح : (d) و (e) و (f) و (k) و (l)

٥ - ٨٣ اكتب صورة مناسبة PICTURE ، واستخدم USAGE إذا لم يكن DISPLAY ، لكل من عناصر البيانات التالية (إذا كان هناك أكثر من خيار واحد ممكن، اكتب أفضلها أولا) .

( a ) رقم الضمان الاجتماعي ( بدون شرطة )

(b) مبلغ فى شيك لاتزيد قيمته عن ١٠٠٠ دولار .

(c) رقم الضمان الاجتماعي ( بشرط ) .

(d) الاسم الأخير (يمكن ان يصل الى ٢٠ رمز) .

(e) عدد ساعات العمل وقت إضافى ( مقربا لاقرب خانة عشرية واحدة ) - ملف شريط .

(f) عدد الخوابير فى المخزن ( يمكن ان يصل الى 99,999 ) - ملف قرص .

(g) موازنة قرض ( حتى ٥٠٠ر٠٠٠ر٠٠ ) - ملف بطاقات .

(h) موازنة قرض ( حتى ٥٠٠ر٠٠٠ر٠٠ ) - ملف قرص .

(i) موازنة قرض ( حتى ٥٠٠ر٠٠٠ر٠٠ ) - تقرير مطبوع .

(j) ملف موازنة قرض ( حتى ٥٠٠ر٠٠٠ر٠٠ ) - تقرير مطبوع .

(k) عدد ساعات العمل وقت اضافى ( لاقرب ١ر٠ ) - ملف بطاقات .

(l) عدد ساعات العمل وقت إضافى (لأقرب ١ر٠) - تقرير مطبوع.

(m) عدد الخوابير فى المخزن ( حتى 99,999 ) - تقرير مطبوع .

(n) تكلفة الشحن للرطل ( لأقرب ١ر٠ سنت ) - ملف بطاقات .

(o) تكلفة الشحن للرطل ( لأقرب ١ر٠ سنت ) - ملف قرص.

(p) تكلفة الشحن للرطل ( لأقرب ١ر٠ سنت ) - تقرير مطبوع .

(q) تاريخ فى الصورة yymmdd ( بدون شرط ) - ملف بطاقات .

(r) تاريخ فى الصورة yymmdd ( بدون شرط ) - ملف قرص .

(s) تاريخ فى الصورة yymmdd ، منقح للطباعة .

(t) شفرة سجل من خانة واحدة - ملف بطاقات .

(u) شفرة سجل من خانة واحدة - ملف قرص .

(v) شفرة سجل من خانة واحدة - تقرير مطبوع .

(w) تعليمات شحن ( حتى ٧٠ رمز ) - ملف بطاقات، وملف قرص، وتقرير طباعة .

(x) رقم جزء ( سبعة أرقام ) - ملف بطاقات .

(y) رقم جزء ( سبعة أرقام ) - ملف قرص .

(z) الراتب الذى ترغب فى الحصول عليه ( ويمكن التعبير عنه بكوبل IBM OS/VS ) - تقرير مطبوع .

٥ - ٨٤ اذكر الطول ، بعدد البايت ، لكل عنصر بيانات معروف فى المسألة السابقة (افترض استخدام نظام IBM OS/VS ، أذكر اطوال كل من COMP و COMP-3 عندما يكون ذلك مناسبا) .

(a) ٩ ( مع DISPLAY ، كل رمز صورة باستثناء S و V يحتاج بايت واحداً) .

(b) ٩ ( مع DISPLAY ، كل رمز صورة باستثناء S و V يحتاج بايت واحداً) .

(c) ١١ ( مع DISPLAY ، كل رمز صورة باستثناء S و V يحتاج بايت واحداً) .

(d) ٢٠ ( مع DISPLAY ، كل رمز صورة باستثناء S و V يحتاج بايت واحداً) .

(e) COMP-3 : 2 [ PIC لها ٣ أرقام ولهذا تأخذ [٣/٢ ] + ١ = ٢ بايت ] .
COMP : ٢ ( يمكنها إمساك من ١ الى ٤ أرقام ) .

(f) COMP-3 : ٣ [ PIC لها ٥ أرقام ولهذا تأخذ [ ٥/٢ ] + ١ = ٣ بايت ] .
COMP : ٤ ( يمكنها إمساك من ٥ الى ٩ أرقام ) .

(g) ٨ ( مع DISPLAY ، كل رمز صورة باستثناء S و V يحتاج بايت واحداً).

(h) COMP-3 : 5 [ PIC لها ٨ أرقام ولهذا تأخذ [ ٨/٢ ] + ١ = ٥ بايت ] .
COMP : ٤ ( يمكنها إمساك من ٥ إلى ٩ أرقام ) .

(i) ١٢

(j) ٣ ( لايحتاج S أو V أى موقع إضافيًا ) .

(k) ٤ ( النقطة تمثل علامة عشرية واقعية وتشغل بايت واحداً) .

(l) ٥

(m) ٧ مع إشارة سالب متخلفة ، أو ٦ بدونها .

(n) ٦ ( لايحتاج S أو V أى موقع إضافيًا ) .

(o) COMP-3 : ٤ [ PIC لها ٦ أرقام ولهذا تأخذ [ ٦/٢ ] + ١ = ٤ بايت ] .
COMP : ٢ ( يمكنها إمساك من ٥ إلى ٩ أرقام ) .

(p) ٧

(q) ٦

(r) ١

(s) ١

(t) ٧

(u) ٦

(v) ٨

١ (w)

٧٠ (x)

٧ (y)

١٧ (z) ( يحتاج كل رمز صورة فى صورة تنقيح إلى بايت واحد) .

٥ – ٨٥ بمعرفة العبارة MOVE ZEROS TO SAMPLE- DATA - ITEM ... بين النتائج عندما يكون SAMPLE- DATA- ITEM معرفا كما يلي :

| | | |
|---|---|---|
| (*a*) PIC S9(3)V99 COMP-3 | (*b*) PIC S9(3)V99 | (*c*) PIC ZZ.ZZ |
| (*d*) PIC $**,***.** | (*e*) PIC $**,***.** BLANK WHEN ZERO | (*f*) PIC Z,ZZZ.99 |
| (*g*) PIC Z,ZZZ.Z9 | (*h*) PIC Z,ZZZ.99 BLANK WHEN ZERO | (*i*) PIC X(4) |
| (*j*) PIC A(4) | (*k*) PIC XX/XX/XX | (*l*) PIC XXBXXBXX |

(a) 00 ‸000 + ( صورة COMP-3 )

(b) 00 ‸000 + ( صورة DISPLAY )

(c) bbbbb

(d) ( $******.** تطبع النقطة دائماً مع حماية الشيكات ).

(e) غير صحيح : لاتستخدم BLANK WHEN ZEROS مع حماية الشيكات .

(f) bbbbb.00

(g) bbbbbbb0

(h) bbbbbbbb

(i) 0000 ( أصفار فى تمثيل EBCDIC )

(j) غير صحيح : يحتوى PIC A على حروف وفراغات فقط .

(k) 00/00/00

(l) 00b00b00

٥ – ٨٦ بمعرفة العبارة التالية MOVE SPACES TO SAMPLE- DATA - ITEM .. بين النتائج عندما يكون SAMPLE- DATA - ITEM معرفا كما يلي :

(*a*) PIC X(3) (*b*) PIC A(4) (*c*) PIC 9(3) (*d*) PIC S9(4)V99
(*e*) PIC XXX/XX

bbb (a)

bbbb (b)

(c) غير صحيح

(d) غير صحيح

bbb/b (e)

( تذكر ان PIC9 لايمكن أن تشتمل إلا على أرقام ) .

٥ – ٨٧ بمعرفة العبارة التالية : MOVE "ABCDE" TO SAMPLE- DATA - ITEM .. بين النتائج عندما يكون SAMPLE- DATA - ITEM معرفًا كما يلى :

(*a*) PIC X(3)  (*b*) PIC X(3) JUSTIFIED RIGHT  (*c*) PIC XX/XX
(*d*) PIC X(8)  (*e*) PIC X(8) JUSTIFIED RIGHT

ABC – a ( الحذف من ناحية اليمين ) .

CDE – b ( الحذف من ناحية اليسار لوجود JUSTIFIED RIGHT ) .

AB/CD – c ( الحذف من ناحية اليمين مع إدخال / ) .

ABCDEbbb – d ( فراغات من ناحية اليمين ) .

bbbABCDE – e (يتسبب جزء JUSTIFIED RIGHT فى نقل القيمة من يمين الحقل، وبالتالى تصبح الفراغات فى يسار الحقل).

٥ – ٨٨ ما أهمية SYNC مع عناصر COMP ؟ تعتمد أهمية استخدام SYNC ، مع عناصر COMP على معمارية نظام الكمبيوتر المستخدم وعلى التنفيذ الفعلى لعناصر COMP . فى نظم كمبيوتر IBM-370 . لايبدو الرمز فى وقت التنفيذ معنويا . بسبب إدخال بايت راكدة ( القسم ١٦ من هذا الفصل ) . يجب إلا يستخدم SYNC مع عناصر COMP الموجودة فى سجلات منطقية بصفة عامة ) .

٥ – ٨٩ ما نتيجة العبارة MOVE ZEROS TO - A- GROUP- ITEM، حيث ان العنصر معرف كما يلى :

```
01  A-GROUP-ITEM.
    05  A          PIC X(2).
    05  B          PIC S9(3)V99.
    05  C          PIC S9(5)V99    COMP-3.
    05  D          PIC S9(4)       COMP.
```

عند معالجة مجموعات عناصر فى جزء الاجراءات فدائما ما تعامل كحقل كبير له صورة من نوع X . وحيث ان طول A- GROUP- ITEM هو ١٣ بايت ( ٢ + ٥ + ٤ + ٢ ) .. فتعامل عبارة MOVE العنصر A- GROUP- ITEM ، كما لو كان معرفا بالصورة PIC X (13) . وعلى هذا.. ينقل ١٣ صفراً من نوع DISPLAY إلى بايت العنصر ، التى تدخل أصفار

صحيحة فى A و B ، حيث إنهما معرفان بانهما DISPLAY ( تقليديا ) . إلا أن C يحتاج صفر من نوع COMP وليست اصفار من نوع DISPLAY. النتيجة هى ان مواقع C و D لاتحتوى على بيانات صحيحة ، وفى الواقع ، معالجة الحقول C ، و D ينتج عنها نتائج غير صحيحة ويمكن أن يفصل تنفيذ البرنامج بنهاية غير طبيعية .

٥ - ٩٠ بمعرفة مايلى :

```
01  SAMPLE-ITEM.
    05  A              PIC  9(4)V99.
    05  B REDEFINES  A.
        10  C          PIC  9(2)V9.
        10  D          PIC  9(3).
```

ما محتويات C , D بعد تنفيذ العبارة التالية :

MOVE 12.34 TO A

| A: PIC 9(4)V99 | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 |
| C: PIC 9(2)V9 | | | D: PIC 9(3) | | |
| B: PIC X(6) | | | | | |

شكل (٥ - ١٠)

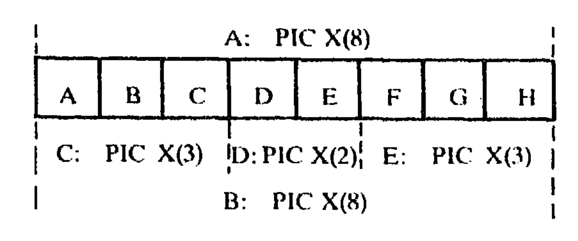
بسبب MOVE يحتوى A على $0012_{\wedge}34$ ، ولكن بسبب REDEFINES تشير D , C إلى نفس مواقع ذاكرة الكمبيوتر تماما مثل A . وهذا الموقف موضح فى شكل ٥ - ١٠ ويمكن رؤية أن C يحتوى النصف الأيسر من A ، وهو ( من نقطة تعريف العنصر C ) 00.1 ، ويحتوى D على ٢٣٤ ( النصف الأيمن من A , B, وهو مجموعة عناصر ، يحتوى على 001234، ويمكن أن يعامل على الصورة PIC X (6) DISPLAY .

٥ - ٩١ بمعرفة مايلى :

```
01  SAMPLE-ITEM.
    05  A              PIC  X(8)     VALUE  "ABCDEFGH".
    05  B REDEFINES  A.
        10  C          PIC  X(3).
        10  D          PIC  X(2).
        10  E          PIC  X(3).
```

حدد ماهية C : فى D ؟ فى E ؟ فى B ؟

حيث إن B يعيد تعريف A .. فإن الموقف يكون موضحا فى شكل (٥ - ١١) .




شكل ( ٥ - ١١ )

نظراً لأن B مجموعة عناصر .. فهو يكافيء PIC X (18) . لاحظ أن فى إعادة تعريف عنصر .. يجب أن تتساوى أطوال العنصرين .C يحتوى على "ABC" و E يحتوى على "FGH" . و B يحتوى على "ABCDEFGH" حيث إنه يشير إلى نفس مواقع الذاكرة تماما مثل العنصر A .

٥ - ٩٢ عرف حقلا يكون جزءا من سطر مطبوع . يحتوى الحقل على (١) مبلغ من المال ( بحد اقصى 50,00.00 ) المربوط فى تخزين عنصر معين، أو (٢) رسالة "STOCK-OUT" إذا لم يوجد هذا العنصر بالشركة .

```
05  VALUE-OF-INVENTORY           PIC ZZ,ZZZ.99.
05  OUT-OF-STOCK-MESSAGE-AREA    REDEFINES VALUE-OF-INVENTORY
                                 PIC X(9).
```

هذه المسألة توضح الحاجة إلى ، ووسيلة الحصول على ، كل من صورة عددية وصورة غير عددية لنفس الحقل من السجل . تتسبب العبارة MOVE "STOCK- OUT" TO VALUE- OF- INVENTORY فى حدوث خطأ تكوينى ، حيث انه يمكن نقل بيانات عددية فقط الى حقل عددى منقح. أما العبارة: MOVE "STOCK- OUT" TO OUT- OF- STOCK- MESSAGE- AREA فهى صحيحة وتحقق ما هو مطلوب بالضبط .

٥ - ٩٣ ما الخطأ فيمايلى :

```
01  SAMPLE-ITEM.
    05  A          PIC X(5).
    05  B          REDEFINES A.
        10  C      PIC S9(5)      COMP-3.
        10  D      PIC S9(4)      COMP.
```

لايوجد شئ خطأ . طول B هو٢ + ٣ + ٥ بايت ، والذى يتفق مع طول A ، كما هو مطلوب . والمبرمج حر فى تغيير تخطيط الحقول الجزئية كلية بالنسبة الى PIC , USAGE وغيرها عندما تستخدم REDEFINES . الشروط الوحيدة هى :

( ١ ) يجب أن يتساوى طول العنصرين .

( ٢ ) لايمكن استخدام REDEFINES مع عناصر المستوى 01 فى قسم الملفات .

( ٣ ) العنصر المعاد تعريفه والعنصر الذى يعيد التعريف.. يجب أن يكون لهما نفس المستوى .

٥ – ٩٤ ماالخطأ فيما يلى :

```
01  SAMPLE-ITEM      PIC X(5).
01  OTHER-ITEM       REDEFINES SAMPLE-ITEM.
    05  A            PIC X(4).
    05  B            PIC X(3).
    04  C            PIC X(2) REDEFINES B.
```

( ١ ) استخدمت REDEFINES مع عناصر على المستوى 01 ، وهذا صحيح فى مخزن العمل إلا أنه خطأ فى قسم الملفات .

( ٢ ) يختلف طول OTHER- ITEM عن طول SAMPLE- ITEM .

( ٣ ) يختلف طول C عن طول B .

( ٤ ) بالنسبة للعنصر C يجب أن يكون REDEFINES هو أول جزء فى وصف العنصر .

( ٥ ) رقم مستوى إعادة تعريف C ليس نفس رقم مستوى العنصر المعاد تعريفه B.

٥ – ٩٥ ما الخطأ فيما يلى :

```
WORKING-STORAGE SECTION.
01  PROGRAM-COUNTERS.
    05  NUMBER-UNPAID-INVOICES    PIC S9(5)    COMP-3.

PROCEDURE DIVISION.

    ADD 1 TO NUMBER-UNPAID-INVOICES
```

تأخذ عبارة ADD المحتويات الحالية للعنصر NUMBER- UNPAID- INVOICES وتضيف ١ ، وتعيد النتيجة فى نفس العنصر . لكن مامحتويات NUMBER- UNPAID- INVOICES الحالية عندما يبدأ تنفيذ البرنامج ؟ حيث إن المبرمج لم يضع قيمة ابتدائية فى عنصر البيانات هذا .. فإن محتوياته ستكون نفايا أو غير معرفة عند تنفيذ ADD، وهذا خطأ شائع جدا بالنسبة للمبرمجين المبتدئين ، وما يتبع ذلك هو عد إجمالى غير صحيح او ABEND ( نهاية غير طبيعية للبرنامج abnormal end) .

٥ – ٩٦ ماذا يمكن عمله لتصحيح الخطأ فى المسألة السابقة ؟

يجب أن توضع قيمة ابتدائية لعنصر البيانات وتكون مساوية صفرا . وحيث ان قيمته تتغير اثناء تنفيذ البرنامج .. فإن الطريقة المفضلة لوضع قيمة ابتدائية، هى باستخدام عبارة MOVE من جزء الإجراءات على النحو التالى :

```
PROCEDURE DIVISION.
    MOVE ZERO TO NUMBER-UNPAID-INVOICES
    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    ADD 1 TO NUMBER-UNPAID-INVOICES
```

حيث إن العنصر NUMBER- UNPAID- INVOICES معرف فى مخزن العمل .. فهناك طريقة بديلة ، وهى استخدام جزء VALUE :

```
05  NUMBER-UNPAID-INVOICES  PIC S9(5)  COMP-3  VALUE ZERO.
```

وهذا يؤكد أن NUMBER- UNPAID- INVOICES يحتوى على صفر ( بدلاً من نفايا ) فى بداية تنفيذ البرنامج . (تحجز بعض نمطيات الكتابة جزء VALUE للعناصر التى لاتتغير قيمها أثناء تنفيذ البرنامج ) .

٥ – ٩٧ لايفضل عديد من المبرمجين كتابة أوصاف تفصيلية للسجلات المنطقية فى قسم الملفات . بدلا من ذلك فإنهم يقومون بتعريف منطقة مخزن عمل يكون لها نفس الحجم مثل السجل المنطقى . وبمجرد ادخال سجل منطقى من ملف ( باستخدام عبارة READ ) .. ينقل البرنامج نسخة من السجل المنطقى إلى داخل منطقة مخزن العمل . وتؤدى كل معالجات البيانات بعد ذلك باستخدام نسخة مخزن العمل . وبالمثل ، توضع سجلات المخرجات مع بعضها فى منطقة مخزن عمل لها حجم مناسب ووصف مناسب ، وينقل بعد ذلك إلى منطقة السجل المنطقى من الملف ( المستوى 01 فى FD ) قبل أن تنفذ عبارة WRITE.

خذ محتويات قسم الملفات التالية وعدلها بحيث يمكن أداء تشغيل السجل فى مخزن العمل .

```
FILE SECTION.

FD  TIME-CARD-INPUT-FILE
    BLOCK CONTAINS 0 CHARACTERS
    RECORD CONTAINS 21 CHARACTERS
    LABEL RECORDS ARE STANDARD

01  TIME-RECORD.
    05  TIME-EMPLOYEE-ID          PIC X(5).
    05  TIME-REGULAR-HOURS        PIC 99V9.
    05  TIME-OVERTIME-HOURS       PIC 99V9.
    05  TIME-EXPENSES             PIC S9(4)V99.
    05  TIME-SICK-LEAVE           PIC 99.
    05  TIME-VACATION             PIC 99.

FILE SECTION.

FD  TIME-CARD-INPUT-FILE
    BLOCK CONTAINS 0 CHARACTERS
    RECORD CONTAINS 21 CHARACTERS
    LABEL RECORDS ARE STANDARD
```

```
01    TIME-RECORD                          PIC X(21).

WORKING-STORAGE SECTION.

01    WORKING-TIME-RECORD-AREA.
      05  WORKING-EMPLOYEE-ID              PIC X(5).
      05  WORKING-REGULAR-HOURS            PIC 99V9.
      05  WORKING-OVERTIME-HOURS           PIC 99V9.
      05  WORKING-EXPENSES                 PIC S9(4)V99.
      05  WORKING-SICK-LEAVE               PIC 99.
      05  WORKING-VACATION                 PIC 99.
```

لاحظ إن وصف السجل فى قسم الملفات اصبح تافها – PIC X (12) فقط ، حيث إن طول السجل هو ٢١ . وقد تم نقل الوصف التفصيلى لكل حقل من حقول السجل الى قسم مخزن العمل . لاحظ أن عناصر البيانات REGULAR-HOURS و OVERTIME- HOURS و SICK- LEAVE لم يحدد لها إشارات عمليات . ولايؤثر هذا على كفاءة برنامج الهدف، طالما ان هذه العناصر لن تستقبل نتائج حسابات ابدا وأنها لن تكون سالبة على الإطلاق ( الشئ الذى يكون مستحيلا ، أن يكون العنصر سالبا ، بدون S ) . وقد يختار المبرمج المحافظ ان يستخدم S بالنسبة لهذه العناصر بغرض الأمان .

٥ – ٩٨ يستخدم منهج مخزن العمل الذى سبق مناقشته فى المسألة السابقة دائما تقريبا فى طباعة ملفات المخرجات، حيث ان سجلاتها تحتوى على معلومات ثابتة كثيرة، مثل عناوين الصفحات ، وعناوين الأعمدة وغيرها . يمكن استخدام جزء VALUE فى مخزن العمل لاعداد مثل هذه السجلات .

راجع المحتويات التالية بحيث تعد السجلات فى مخزن العمل . انتج كذلك سجلا لطباعة عناوين أعمدة فوق المعلومات المبينة.

```
FILE SECTION.

FD    PAYROLL-REGISTER-FILE
      RECORD CONTAINS 132 CHARACTERS
      LABEL RECORDS ARE OMITTED

01    PAYROLL-REGISTER-LINE.
      05  REGISTER-EMPLOYEE-ID             PIC X(5).
      05  FILLER                           PIC X(10).
      05  REGISTER-REGULAR-HOURS           PIC ZZ.9.
      05  FILLER                           PIC X(10).
      05  REGISTER-OVERTIME-HOURS          PIC ZZ.9.
      05  FILLER                           PIC X(99).

FILE SECTION.

FD    PAYROLL-REGISTER-FILE
      RECORD CONTAINS 132 CHARACTERS
      LABEL RECORDS ARE OMITTED
```

```
01   PAYROLL-REGISTER-LINE                 PIC X(132).

WORKING-STORAGE SECTION.

01   WORKING-REGISTER-LINE.
     05  WORKING-LINE-EMPLOYEE-ID          PIC X(5).
     05  FILLER                            PIC X(10)   VALUE SPACES.
     05  WORKING-LINE-REGULAR-HOURS        PIC ZZ.9.
     05  FILLER                            PIC X(10)   VALUE SPACES.
     05  WORKING-LINE-OVERTIME-HOURS       PIC ZZ.9.
     05  FILLER                            PIC X(99)   VALUE SPACES.

01   WORKING-HEADING-LINE.
     05  FILLER                            PIC X(15)
                                           VALUE "EMPLOYEE ID".
     05  FILLER                            PIC X(14)
                                           VALUE "REGULAR HOURS".
     05  FILLER                            PIC X(103)
                                           VALUE "OVERTIME HOURS".
```

لاحظ ان PAYROLL- REGISTER- LINE معرف الآن ببساطة بواسطة PIC X (132) ، مع عمل كل الاوصاف الفعلية فى WORKING- REGISTER- LINE . يمكن أن توضع فراغات فى مناطق FILLER ، لفصل البيانات المطبوعة فى السطر ، وذلك باستخدام جزء VALUE ( والذى لايمكن استخدامه عندما يحدث الوصف فى قسم الملفات ) . لاتوجد فى WORKING- HEADING- LINE الا محتويات FILLER ولها اجزاء VALUE وذلك لانتاج عناوين الاعمدة المناسبة . اقنع نفسك ان عنوان كل عمود يبدأ فوق اول رمز معلومات للعمود . ( لاحظ ، على أية حال ، أنها ليست متمركزة) .

## تمـــارين برمجـــة

من ٥ – ٩٩ إلى ٥ – ١٠٨ عدل التمارين من رقم ٢ – ٢٩ إلى رقم ٢ – ٣٨: لتعكس معلوماتك الموسعة عن جزء البيانات . غير كل القيم الدولارية إلى دولارات وسنتات ، مع أجراء أى تعديلات على تخطيطات السجلات إذا كانت هناك حاجة لذلك . نقح كل المخرجات المطبوعة ، وأجر الحسابات بالكفاءة الممكنة .

الفصل السادس

# جــزء الإجــراءات

# Procedure Division

## ٦ - ١ مقدمة : نمطيات كتابة الشفرة

تخدم أجزاء التعريف والأوساط والبيانات كنوع من المقدمة لجزء الإجراءات ، والذى يحدد فيه المبرمج التعليمات التى يؤديها الكمبيوتر . ويفترض فى هذا الفصل أن هذه التعليمات تعمل مع مدخلات وتشغيل ومخرجات الملفات المرتبة تتابعيا (التى يتم الاتصال بها تتابعيا) .

وهيكليا .. يتكون جزء الإجراءات من اقسام SECTIONS ومقاطع . ويكتب عنوان الجزء "PROCEDURE DIVI-SION" فى المنطقة A ، كما تبدأ كتابة عناوين الأقسام والمقاطع فى المنطقة A كذلك . وتتكون الأقسام والمقاطع من جمل sentences والتى تتحدد بنقاط . يمكن أن تتكون كل جملة من عبارة statement واحدة أو أكثر ، والعبارة هى احدى التعليمات الصحيحة من جزء الإجراءات، التى تبدأ بفعل verb من أفعال الكوبل .

## نمطيات كتابة جزء الإجراءات

( ١ ) اكتب عبارة واحدة فقط فى السطر .

( ٢ ) يجب أن تخصص سطور لعناوين المقاطع والأقسام .

( ٣ ) يجب ان يكون لكل مقطع وكل قسم غرض معرف تعريفا جيدا ، ويجب أن يصف الاسم هذا الغرض .

( ٤ ) رتب عناوين المقاطع والأقسام باضافة سابقات او لاحقات للأسماء .

( ٥ ) أعمل ترحيل ( عدد معين من الخانات بدءا من العمود الثانى عشر ) ؛ لتوضيح استمرار العبارة فى أكثر من سطر واحد، وتوضيح العلاقات الموجودة بين العبارات .

( ٦ ) استخدم أسطر فارغة ، أسطر تعليق فارغة ، أو استخدم SKIP3, SKIP2 , SKIP1 , EJECT ( وهى أفضل إذا كانت متاحة ) فى ترك فراغ بين المقاطع والأقسام وتحديد العناوين .

( ٧ ) استخدم النقط عندما تكون هناك فقط ضرورة لها ، واجعلها نقطًا مرتبة .

( ٨ ) استخدم فواصل عندما تكون هناك فقط ضرورة لها .

( ٩ ) تجنب استخدام أقسام الا إذا كان هذا مطلوبا .

(١٠) اجعل كل شئ بسيطًا ومباشرًا بقدر الإمكان .

## ٦ - ٢ مدخلات/ مخرجات : عبارة الفتح

يجب أن يفتح OPEN كل ملف بنجاح قبل استخدام أى عبارة من عبارات جزء الإجراءات فى تشغيل الملف :

( ١ ) تحصل عبارة الفتح على ذاكرات احتياطية للملف من مواقع الذاكرة الرئيسية للملف .

( ٢ ) وتقوم بتشغيل عناوين الملف إذا كانت موجودة ( إذا كانت LABEL RECORDS ARE STANDARD محددة فى FD الملف ) .

- **ملف مدخلات، أو ملف مدخلات و مخرجات** : تختبر سجلات العناوين الموجودة؛ للتأكد من أن كل المعلومات متاحة، ان الملف يمكن الاتصال به، وأنه معرف تعريفا صحيحا.
- **ملف مخرجات** : يتم إنتاج سجلات عناوين للملف .

( ٣ ) ويعد الملف للاتصال بسجلاته المنطقية - عادة تجعله عند بدايته ( أى عند بداية أول سجل منطقى ) .

التكوين العام لعبارة الفتح OPEN فى شكل (٦ - ١) . كما تحدد الأقواس.. يجب أن تحدد إحدى حالات OPEN الأربع على الأقل ، بعدد الملفات المراد فتحها فى كل حالة .

```
       { INPUT   file-name-1 [REVERSED       ]                        }
       {                     [WITH NO REWIND ]                        }
       {         [file-name-2 [REVERSED      ]] ...                   }
       {                      [WITH NO REWIND]                        }
OPEN   { OUTPUT  file-name-3 [WITH NO REWIND]                         } ...
       {         [file-name-4 [WITH NO REWIND]] ...                   }
       { I-O file-name-5 [file-name-6] ...                            }
       { EXTEND file-name-7 [file-name-8] ...                         }
```

شكل (٦ - ١)

- **حالة الإدخال** INPUT : يفتح الملف للإدخال فقط ، ويجب أن يكون الملف موجودا فعلا ويمكن الاتصال به ، وإلا فيفصل البرنامج فصلا غير طبيعى . ويستخدم فعل القراءة READ ، وهو الفعل الوحيد المستخدم ، فى إدخال سجل منطقى.
- **حالة الإخراج** OUTPUT : يفتح الملف للإخراج فقط ، والفعل الوحيد المستخدم هو فعل الكتابة WRITE . إذا كان الملف موجودا فعلا ، فتحذف محتوياته القديمة وتنتج محتوياته الجديدة باستخدام أمر الكتابة فى إخراج السجلات المنطقية . فإذا لم يكن الملف موجودا ، فيتم إنتاج الملف باستخدام WRITE فى إخراج السجلات المنطقية.
- **حالة الإدخال والإخراج** I - O : يفتح الملف لكل من الإدخال والإخراج ، ويجب أن يكون الملف موجودا ويمكن

الاتصال به وإلا انتهى البرنامج نهاية غير طبيعية . والأفعال المستخدمة هنا هى REWRITE , WRITE , READ، بالنسبة لملف على وحدة اتصال مباشر ، ويمكن استخدام REWRITE فى كتابة صيغة معدلة من سجل منطقى فى نفس موقع السجل الاصلى من الملف ( التجديد فى نفس الموقع ) .

- **حالة موسعة** EXTEND : يفتح الملف للمخرجات فقط ، مع تحديد وضعه فى نهاية الملف ( بعد آخر سجل منطقى ) . وعلى هذا ، عندما تستخدم عبارة WRITE فى إخراج سجلات منطقية ، فإنها تضيف إلى النهاية الواقعية للملف بعد السجلات الموجودة فعلا فى الملف .

ونادرا.. مايستخدم الخياران WITH NO REWIND, REVERSED، وهما يستخدمان مع الملفات الموجودة على شريط مغناطيسى فقط .

- **خيار العكس** REVERSED : يمكن استخدام العكس مع ملفات الشرائط المفتوحة كمدخلات فقط ، ويحدد هذا الجزء أن الملف يفتح عند آخر سجل منطقى مع القراءة العكسية ، أى من الخلف إلى الأمام ( ولاتوجد هذه الإمكانية فى كل مشغلات الشرائط ) .

- **خيار** WITH NO REWIND : يمكن استخدام هذا الخيار ( دون اعادة اللف ) مع ملفات الشرائط المفتوحة كمدخلات فقط أو كمخرجات فقط . وهو خيار يحدد أن الملف يجب إلا يتغير وضعه أثناء عملية فتحه . وعلى هذا يظل الملف فى نفس موقعه عندما تنفذ عبارة الفتح .

**مثـال ٦ - ١ :**

يجب أن تناظر كل أسماء الملفات المستخدمة فى عبارة الفتح التالية الأسماء المستخدمة فى عبارة SELECT ، وفى FD .

```
OPEN    INPUT     EDITED-TIME-FILE
                  EMPLOYEE-MASTER-FILE-OLD
        OUTPUT    PAYROLL-REGISTER-REPORT
                  EMPLOYEE-MASTER-FILE-NEW
        EXTEND    TIME-RECORD-SUMMARY-FILE
```

لاحظ نمطية الكتابة هذه لعبارة الفتح : كتبت كل أسماء الملفات فى نفس العمود ، وكتبت كل حالات OPEN فى نفس العمود، وذلك مع استخدام الترحيل المناسب .

وكبديل.. يمكن أن يكون لكل ملف عبارة OPEN خاصة به . وبصفة عامة .. فإن فتح عديد من الملفات بعبارة فتح واحدة يأخذ ذاكرة كمبيوتر أكبر لتشغيل OPEN ، ولكنه - فى نفس الوقت - يسمح باتمام الفتح بشكل أسرع كثيراً .

**مثـال ٦ - ٢ :**

مايلى يوضح خطأ شائعًا يقع فيه المبرمجون المبتدئون .

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  MISTAKE.

ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT TIME-CARD-FILE
        ASSIGN TO WKLYTIME
```

```
    SELECT PAYROLL-EDIT-REPORT
        ASSIGN TO TIMEEDIT

DATA DIVISION.
FILE SECTION.
FD  TIME-CARD-FILE
    LABEL RECORDS ARE OMITTED
    .
01  TIME-CARD-RECORD.
    05 TIME-CARD-ID              PIC X(6).
    . . . . . . . . . . . . . .

FD  PAYROLL-EDIT-REPORT
    LABEL RECORDS ARE OMITTED
    RECORD CONTAINS 132 CHARACTERS
    .
01  EDIT-LINE.
    05  EDIT-ID                  PIC X(6).
    05  FILLER                   PIC X(5).
    . . . . . . . . . . . . . .

PROCEDURE DIVISION.
    MOVE ALL "*" TO EDIT-LINE
    WRITE EDIT-LINE
        AFTER ADVANCING PAGE
    OPEN    OUTPUT    PAYROLL-EDIT-REPORT
    OPEN    INPUT     TIME-CARD-FILE
    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

افحص بدقة جزء الإجراءات، منطقة السجل المنطقى EDIT- LINE الخاصة بـ REPORT،PAYROLL- EDIT- يجب إلا تعالج أو تستخدم بعد فتح الملف . وكما تبدو الأشياء ستكون للعبارة MOVE ALL "*" TO EDIT- LINE نتائج غير متوقعة ، وتتسبب WRITE EDIT- LINE فى إنهاء البرنامج نهاية غير طبيعية .

**مثـال ٦ – ٣ :**

افرض أن جزء FILE STATUS ( القسم الخامس من الفصل الرابع ) مستخدم فى البرنامج ( المصحح ) من مثال (٦–٢).

```
. . . . . . . . . . . . . . . . . . . . . . . . .
FILE-CONTROL.
    SELECT TIME-CARD-FILE
        ASSIGN TO WKLYTIME
        FILE STATUS IS TIME-CARD-STATUS

    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
WORKING-STORAGE SECTION.
01  TIME-CARD-STATUS             PIC XX.
. . . . . . . . . . . . . . . . . . .
PROCEDURE DIVISION.
. . . . . . . . . . . . . . . . . . .
    OPEN INPUT TIME-CARD-FILE
    IF TIME-CARD-STATUS NOT EQUAL "00"
    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

TIME- CARD- STATUS معرف بأنه منطقة مكونة من ٢ بايت (PIC XX) . بعد تنفيذ عبارة الفتح يجب أن تحتوى هذه المنطقة الثابت غير العددى "00" ، اذا كان التنفيذ صحيحا . أما إذا كان TIME- CARD- STATUS معرفا بأنه PIC 99 .. فإنه يستخدم على ذلك ثابتًا عدديًا ( أو ثابتًا استعاريًا مكافئًا له ) فى الاختيار :

```
IF TIME-CARD-STATUS NOT EQUAL 0 ...
```

أو

```
IF TIME-CARD-STATUS NOT EQUAL ZERO ...
```

## ٦ - ٣ مدخلات / مخرجات : عبارة الإغلاق

تنهى عبارة الاغلاق CLOSE التشغيل للملف وذلك ( ١ ) بترك موقع الذاكرة المستخدم للذاكرات الاحتياطية . ( ٢ ) واتمام تشغيل عناوين ضرورى للملفات التى لها نمطية ( ٣ ) وعدم عمل كل شئ فعلته عبارة الفتح بصفة عامة . لابد من إغلاق كل ملف سبق فتحه قبل إنهاء البرنامج ، ونسيان ذلك يمكن أن يقود إلى أخطاء جسيمة فى بعض النظم . وتكوين عبارة الإغلاق موضح فى شكل (٦ - ٢) .

```
                    ┌ {REEL}  [WITH NO REWIND] ┐
                    │ {UNIT}  [FOR REMOVAL   ] │
CLOSE file-name-1   │                          │
                    │  WITH  {NO REWIND}       │
                    └        {LOCK     }       ┘
                    ┌ {REEL}  [WITH NO REWIND] ┐
                    │ {UNIT}  [FOR REMOVAL   ] │
  [file-name-2      │                          │ ] ...
                    │  WITH  {NO REWIND}       │
                    └        {LOCK     }       ┘
```

شكل (٦ - ٢)

## الصيغة الأساسية للإغلاق

الصيغة الأكثر استخداما لعبارة الإغلاق هى – ببساطة – الفعل يتبعه اسم ملف واحد، أو أسماء ملفات . يمكن غلق الملفات بشكل فردى ( مع أخذ ذاكرة أقل من الكمبيوتر ) أو مع بعضها ( مع أستغراق وقت أقل ) ، وبأى ترتيب ( دون الاعتماد على ترتيب فتح الملفات )، وأيا كانت الصيغة المستخدمة لعبارة الإغلاق ، إذا حدد FILE STATUS للملف، الذى يغلق.. فإن عبارة الإغلاق تتسبب فى وضع شفرة حالة فى عنصر بيانات FILE STATUS ، محددة بذلك نجاح أو فشل محاولة الإغلاق .

**مثـال (٦ - ٤) :**

```
PROCEDURE DIVISION.
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    OPEN    OUTPUT    INVENTORY-MASTER-FILE
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
       (use WRITE to add records to INVENTORY-MASTER-FILE)
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    CLOSE INVENTORY-MASTER-FILE
    OPEN    INPUT     INVENTORY-MASTER-FILE
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
       (use READ to input logical records from INVENTORY-MASTER-FILE)
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    CLOSE INVENTORY-MASTER-FILE
    OPEN    I-O       INVENTORY-MASTER-FILE
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
       (use READ and REWRITE to update logical records from INVENTORY-MASTER-FILE)
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    CLOSE INVENTORY-MASTER-FILE
```

بمجرد إغلاق الملف.. يمكن إعادة فتحه للتشغيل مرة أخرى . وهنا تم فتح ملف للمخرجات ، وأضيفت بعض السجلات ، ثم أغلق الملف . بعد ذلك فتح الملف للمدخلات ، واسترجعت السجلات المنطقية ( مثلما يحدث فى طباعة تقرير ) ، ثم أغلق مرة ثانية . وأخيرا .. فتح الملف مرة أخرى للمدخلات والمخرجات، وتم تجديد بعض السجلات المنطقية ، وأغلق الملف بعد ذلك .

**مثـال ٦ - ٥ :**

```
PROCEDURE DIVISION.
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    OPEN  I-O  EMPLOYEE-MASTER-FILE
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    CLOSE  EMPLOYEE-MASTER-FILE    WITH LOCK
```

عندما يستخدم جزء CLOSED WITH LOCK .. يغلق الملف، ولايمكن فتحه أثناء التشغيل الحالى للبرنامج . ويمكن، بالطبع ، فتح الملف بواسطة برنامج آخر ، أو بنفس البرنامج إذا أعيد تنفيذه .

## صيغ مصغرة للإغلاق

تستخدم هذه الصيغ مع ملفات الشرائط فقط . وتسمى بكرة الشريط المخزن عليها أكثر من ملف واحد ببكرة متعددة multiple reel (أو شريط متعدد multiple tape أو حجم متعدد multiple volume) . وتحمل عنوان حجم volume label والذى يعطى اسما للبكرة (اسم الحجم) ومعلومات أخرى عن محتويات الشريط . ولكل ملف عنوانان ملف file labels: عنوان أمامى header label فى بداية الملف، وعنوان خلفى trailer label فى نهايته . ويظهر العنوان الخلفى فى حالة تشغيل الملف عكسيا (OPEN INPUT REVERSED) .

وعكسيا ، فالملف متعدد الحجوم multivolume file هو ملف مخزن على أكثر من بكرة واحدة ( أو أكثر من مجموعة أقراص واحدة ) .

**مثـال ٦ - ٦ :**

```
CLOSE   EMPLOYEE-MASTER-FILE   WITH NO REWIND
```

عندما يغلق ملف شريط ، - عادة - يعاد لف الشريط الى بداية الملف ، وهنا ( ١ ) يوجد EMPLOYEE- MASTER- FILE على بكرة متعددة الملفات ، ونريد ترك الشريط فى موضع فتح الملف التالى على الشريط، أو ( ٢ ) يعاد فتح الملف EM- PLOYEE- MASTER- FILE للإدخال العكسى INPUT REVERSED ، ويراد تركه موجوداً عند نهايته .

**مثـال ٦ - ٧ :**

```
PROCEDURE DIVISION.
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    OPEN     OUTPUT    JOB-COSTING-HISTORY-FILE
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    WRITE JOB-COSTING-HISTORY-RECORD
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    CLOSE JOB-COSTING-HISTORY-FILE      UNIT
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    WRITE JOB-COSTING-HISTORY-RECORD
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    CLOSE JOB-COSTING-HISTORY-FILE
```

ينتج هذا البرنامج ملف JOB- COSTING- HISTORY- FILE وهو متعدد الحجوم ، سبق فتحه كملف مخرجات . تستخدم أول عبارة WRITE فى إخراج سجلات إلى الملف . عبارة CLOSE... UNIT ( أو المكافئ لها تماما CLOSE... REEL ) لاتغلق الملف فعليا ، الا أنها تتأكد تلقائيا أن عبارة WRITE التالية ستخرج سجلا إلى البكرة أو مجموعة الاقراص التالية من الملف ( يسمى هذا بمفتاح الحجم volume switch ) . لاحظ أنه عندما كتبت كل السجلات المنطقية.. تم أغلاق الملف بطريقة طبيعية .

بالنسبة للملف متعدد الحجوم الذى فتح كمدخلات.. تدخل عبارة READ - التى تاتى بعد CLOSE ...UNIT - سجلا من البكرة التالية أو مجموعة الاقراص التالية .

إذا استبدلت عبارة الإغلاق ، فى البرنامج السابق ، بالعبارة التالية :

```
CLOSE JOB-COSTING-HISTORY-FILE      UNIT      FOR REMOVAL
```

فلن يؤثر نظام التشغيل فقط على مفتاح الحجم فقط، ولكنه يعرف كذلك أنه ليست هناك حاجة إلى الوحدة التى أغلقت مرة أخرى. ويمكن أن يحرر ذلك وحدة المدخلات والمخرجات لاستخدامها فى أى استخدام آخر . إلا أنه يمكن الاتصال بالوحدة المغلقة مرة أخرى، إذا أغلق الملف بطريقة طبيعية ثم أعيد فتحه .

## عبارات مدخلات ومخرجات اخرى

بجانب عبارتى الفتح OPEN والإغلاق CLOSE ، توجد عبارة إعادة الكتابة REWRITE ، والتى يمكنك الرجوع اليها فى الفصل الحادى عشر من الكتاب .

## ٦ - ٤ مدخلات : عبارة القراءة

يمكن استخدام عبارة القراءة مع الملفات التى تم فتحها - بنجاح - كمدخلات او مخرجات فقط . وشكلها العام هو :

```
READ file-name RECORD [INTO data-name]
     [AT END imperative-statement-1 [imperative-statement-2] . . .]
```

يجب أن يكون file- name هو نفسه المستخدم فى عبارة SELECT وفى FD للملف . بافتراض الترتيب التتابعى - كما هو الحال دائما - تجعل عبارة READ السجل المنطقى التالى متاحا واقعيا فى منطقة السجل المنطقى للملف ( المستوى 01 لوصف السجل فى FD ) . فاذا ماعرفت منطقة FILE STATUS للملف.. تضع عبارة READ محتويات مفتاح الحالة لتحديد نتائج READ .

**مثـال ٦ - ٨ :**

يتطلب تكوين READ استخدام اسم الملف بدلا من اسم السجل المنطقى . وعكس ذلك تماماً صحيح بالنسبة لعبارة WRITE ( انظر القسم الخامس من هذا الفصل ) . ولتجنب الخلط .. تذكر شعار الكويل « اقرأ ملفًا واكتب سجلاً » .

عندما تنفذ READ ولاتكون هناك سجلات منطقية متاحة ، يتحقق شرط نهاية الملف end of file . وتنفذ العبارات الموجودة فى جزء AT END وذلك عندما يتحقق شرط نهاية الملف فقط ، وإلا أهمل الكمبيوتر هذا الجزء . عند الوصول إلى نهاية الملف.. يجب اغلاق الملف والاستجابة قبل محاولة أى تشغيل آخر .

**مثـال ٦ - ٩ :**

```
. . . . . . . . . . . . . . . . . . . . . . . . . . . .
ENVIRONMENT DIVISION.

    SELECT SALES-FILE   ASSIGN TO SALES.

DATA DIVISION.
FILE SECTION.

FD  SALES-FILE
        BLOCK CONTAINS 0 RECORDS
        RECORD CONTAINS 39 CHARACTERS
        LABEL RECORDS ARE STANDARD


01  SALES-RECORD.
        05  SALES-RECORD-NAME                 PIC X(15).
        05  SALES-RECORD-MONTH-1-SALES        PIC S9(4)V99.
        05  SALES-RECORD-MONTH-2-SALES        PIC S9(4)V99.
        05  SALES-RECORD-MONTH-3-SALES        PIC S9(4)V99.
        05  SALES-RECORD-QUOTA                PIC S9(4)V99

WORKING-STORAGE SECTION.

01  WORK-AREAS.
        05  SALES-FILE-END-SW                 PIC X.
        05  QUARTERLY-TOTAL                   PIC S9(5)V99 COMP-3.
```

```
PROCEDURE DIVISION.

OPEN  INPUT  SALES-FILE
MOVE "F" TO SALES-FILE-END-SW
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
READ SALES-FILE
    AT END
        MOVE "T" TO SALES-FILE-END-SW
.
IF SALES-FILE-END-SW EQUAL "F"
    ADD  SALES-RECORD-MONTH-1-SALES
         SALES-RECORD-MONTH-2-SALES
         SALES-RECORD-MONTH-3-SALES
                                        GIVING QUARTERLY-TOTAL

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

بعد فتح SALES- FILE كمدخلات ، توضع قيمة ابتدائية للمفتاح SALES- FILE- END- SW مساوية "F" (أى خطأ fa lsc) . يحاول بعد ذلك البرنامج قراءة سجل منطقى SALES- FILE .

فإذا لم تتواجد سجلات منطقية ينفذ AT END ، ويوضع "T" ( اى صحيح truc ) كقيمة للمفتاح . فإذا ما تمت قراءة سجل بنجاح فعلا ، يهمل جزء AT END ويظل المفتاح له القيمة "F" . لاحظ أن البرنامج يختبر ماإذا كان SALES- FILE فى حالة AT END قبل محاولته تنفيذ ADD . لاحظ كذلك استخدام النقطة فى تحديد تعليق لجزء AT END . هذا ضرورى، حيث أنه يفترض أن يستمر الجزء حتى تظهر نقطة .

**مثال ٦ – ١٠ :**

اعتبر جزء البرنامج التالى :

```
READ SALES-FILE RECORD
    AT END
        MOVE "T" TO SALES-FILE-END-SW
MOVE SALES-RECORD-QUOTA TO SALES-REPORT-QUOTA
MOVE "NO" TO SALES-INPUT-ERROR-FLAG
WRITE...
```

أولا.. نلاحظ أن المبرمج ، بنسيانه وضع نقطة بعد جزء AT END استخلص ...MOVE و ...MOVE و ...WRITE غير كفؤة قبل نهاية الملف ( الترحيل لم يقاوم ) . ثانيا.. وربما أقل وضوحا ، لقد نسى أنه إذا قابلت READ نهاية الملف.. فإن منطقة السجل المنطقى SALES- RECORD ستحتوى على نفايا ، تنقل على ذلك إلى SALES- REPORT- QUOTA : متسببة فى مخرجات غير صحيحة وربما فى أخطاء أخرى .

لاتقم بتشغيل سجل منطقى بعد تنفيذ ...AT END (إلا إذا كان الملف قد أغلق وأعيد فتحه ) .

## بديل اقرأ ... فى

يتسبب بديل اقرأ ... فى READ... INTO لعبارة ERAD فى نسخ السجل المنطقى الذى يكون مدخلات من منطقة السجل المنطقى فى عنصر بيانات آخر ؛ أى أن :

```
READ SALES-FILE INTO WORKING-COPY
```

و

```
READ SALES-FILE
MOVE SALES-RECORD TO WORKING-COPY
```

متكافئتان تماما ، وتستخدم READ... INTO بكثرة لتوفير نسخ من السجلات المنطقية فى مخزن العمل ( انظر المسألتين ٩٧ و ٩٨ من الفصل الخامس ) .

ويمكن تطبيق هذا البديل فى كوبل 1974 النمطى على سجلات ثابتة الطول فقط ( انظر ملحق جـ بالنسبة لكوبل 1985 النمطى ).

## ٦ - ٥ مخرجات : عبارة الكتابة

يمكن استخدام عبارة الكتابة WRITE مع الملفات التى سبق فتحها كملفات مخرجات، أو ملفات مدخلات ومخرجات أو ملفات موسعة . ينقل تنفيذ العبارة محتويات منطقة السجل المنطقى ( المستوى 01 ) إلى المخرجات ، وفى نفس الوقت يغير الاشارة إلى اسم السجل على مستوى 01 إلى منطقة الذاكرة الاحتياطية غير المستخدمة حاليا المصاحبة للملف .

(راجع «الذاكرات الاحتياطية المتعددة» فى القسم الثامن من الفصل الخامس و«التجميع» فى القسم الثالث من الفصل الخامس ، ويمكن استخدام جزء الذاكرة الاحتياطية الجديدة ، والذى يحتوى على نفايا عند هذه النقطة ، فى بناء السجل المنطقى التالى للمخرجات)، ويتبع ذلك ، أن السجل المنطقى الذى كتب توًا لايمكن الاتصال به بواسطة البرنامج .

## الكتابة لملفات غير موجهة للطابع

يكون تكوين عبارة WRITE هنا على النحو التالى :

```
WRITE logical-record-name [FROM data-name]
```

يستخدم المبرمج جزء FROM عادة، وعندما يريد عمل كل التشغيل للسجل، مستخدما نسخا للسجل المنطقى من مخزن العمل . ( تناظر READ... FROM من القسم الرابع من هذا الفصل ) .

```
WRITE INVOICE-TO-BE-PAID-RECORD
    FROM WS-INVOICE-TO-BE-PAID-REC
```

لها نفس ترجمة الآلة تماما مثل :

```
MOVE WS-INVOICE-TO-BE-PAID-REC
    TO INVOICE-TO-BE-PAID-RECORD
WRITE INVOICE-TO-BE-PAID-RECORD
```

يجب أن يكون السجل المنطقى logical- record- name اسم العنصر الموجود على مستوى 01 فى FD للملف .

عند استخدام جزء FROM ، عادة مايعرف "data- name" فى مخزن العمل، وإذا تم تعريف منطقة FILE STATUS للملف.. فإن WRITE تضع شفرة مكونة من 2 بايت فى منطقة مفتاح الحالة لتحديد نتيجة تنفيذ WRITE

مثـال ٦ - ١١ :

```
FD  PENDING-INVOICE-FILE
    BLOCK CONTAINS 0 RECORDS
    RECORD CONTAINS 32 CHARACTERS
    LABEL RECORDS ARE STANDARD
    .
01  PENDING-INVOICE-RECORD           PIC X(32).

FD  PAYABLE-INVOICE-FILE
    BLOCK CONTAINS 0 RECORDS
    RECORD CONTAINS 16 CHARACTERS
    LABEL RECORDS ARE STANDARD
    .
01  PAYABLE-INVOICE-RECORD           PIC X(16).

WORKING-STORAGE SECTION.

01  WS-PENDING-INVOICE REC.
    05 WS-PENDING-CODE                PIC X.
    05 WS-PENDING-VENDOR-ID           PIC X(7).
    05 WS-PENDING-INVOICE-NUMBER      PIC X(4).
    05 WS-PENDING-INVOICE-DATE        PIC 9(6).
    05 WS-PENDING-INVOICE-DUE-DATE    PIC 9(6).
    05 WS-PENDING-INVOICE-AMOUNT      PIC S9(5)V99  COMP-3.
    05 WS-PENDING-INVOICE-DISCOUNT    PIC S9(5)V99  COMP-3.

01  WS-PAYABLE-INVOICE-REC.
    05 WS-PAYABLE-CODE                PIC X.
    05 WS-PAYABLE-VENDOR-ID           PIC X(7).
    05 WS-PAYABLE-INVOICE-NUMBER      PIC X(4).
    05 WS-PAYABLE-AMOUNT              PIC S9(5)V99  COMP-3.

PROCEDURE DIVISION.

    OPEN   INPUT    PENDING-INVOICE-FILE
           OUTPUT   PAYABLE-INVOICE-FILE
    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    READ PENDING-INVOICE-FILE
       INTO WS-PENDING-INVOICE-REC
       AT END
          MOVE "T" TO WS-END-OF-FILE-SW
    .
    IF WS-END-OF-FILE-SW NOT EQUAL "T"
       MOVE WS-PENDING-CODE             TO WS-PAYABLE-CODE
       MOVE WS-PENDING-VENDOR-ID        TO WS-PAYABLE-VENDOR-ID
       MOVE WS-PENDING-INVOICE-NUMBER   TO WS-PAYABLE-INVOICE-NUMBER
     SUBTRACT WS-PENDING-INVOICE-DISCOUNT
```

```
        FROM WS-PENDING-INVOICE-AMOUNT
          GIVING WS-PAYABLE-AMOUNT

    WRITE PAYABLE-INVOICE-RECORD FROM WS-PAYABLE-INVOICE-REC

.................................................................
    CLOSE PENDING-INVOICE-FILE
          PAYABLE-INVOICE-FILE
.................................................................
```

يوضح هذا البرنامج استخدام نسخ مخزن العمل للسجلات المنطقية .

يوصف PENDING- INVOICE- RECORD و PAYABLE- INVOICE- RECORD أكثر بصورة PIC X (33) و PX (16) على التوالى، حيث أنه لن يجرى عليهما تشغيل فى منطقة السجل المنطقى للذاكرة الاحتياطية . ويستخدم READ PENDING- INVOICE- FILE INTO WS- PENDING- INVOICE- REC فى إدخال سجل منطقى ونقل نسخة منه الى مخزن العمل . وتستخدم عبارات MOVE و SUBTRACT فى اعداد السجل المنطقى فى WS- PAYABLE- IN- VOICE- REC . ثم تستخدم بعد ذلك العبارة WRITE PAYABLE- INVOICE- RECORD FROM WS- PAYABLE- INVOICE- REC فى نقل هذا السجل من مخزن العمل إلى منطقة السجل المنطقى، فى الذاكرة الاحتياطية ونقله إلى المخرجات .

بعد عبارة WRITE.. لايحتوى PAYABLE- INVOICE- RECORD على البيانات التى أخرجت على التو، بل أنه يشير إلى جزء الذاكرة الاحتياطية التالى الفارغ والمتاح ، والذى يمكن استخدامه فى إعداد السجل المنطقى الجديد . إلا أن WS- PAYABLE- INVOICE- REC لايزال محتويا على السجل المنطقى الذى كتب على التو ، ويشتمل هذا على ميزة أخرى؛ تتعلق بإجراء تشغيل كل السجلات فى مخزن العمل .

كما فى استخدام READ... INTO يجب إلا تستخدم WRITE... FROM مع سجلات متغيرة الأطوال على الاطلاق (انظر ملحق حـ لكوبل 1985 النمطى) . يتطلب منهج مخزن العمل لمثل هذه السجلات زوجا من عبارات MOVE و WRITE :

```
    MOVE WS-PAYABLE-INVOICE-REC TO PAYABLE-INVOICE-RECORD
    WRITE PAYABLE-INVOICE-RECORD
```

## الكتابة لملفات موجهة للطابع

يبين تكوين عبارة WRITE فى هذه الحالة فى شكل (٦ - ٣) .

```
WRITE record-name [FROM identifier]
                               ⎧ ⎧identifier-2⎫ ⎡LINE ⎤ ⎫
  ⎡ ⎧BEFORE⎫                   ⎪ ⎩integer     ⎭ ⎣LINES⎦ ⎪ ⎤
  ⎢ ⎨      ⎬ ADVANCING         ⎨                        ⎬ ⎥
  ⎣ ⎩AFTER ⎭                   ⎪ ⎧mnemonic-name⎫        ⎪ ⎦
                               ⎩ ⎩PAGE         ⎭        ⎭
[AT ⎧END-OF-PAGE⎫ imperative-statement-1 [imperative-statement-2] . . .]
    ⎩EOP        ⎭
```

(WRITE, FROM, BEFORE, AFTER, ADVANCING keywords underlined in original: WRITE, FROM, BEFORE, AFTER, PAGE, END-OF-PAGE, EOP)

شكل (٦ - ٣)

مثـال (٦ - ١٢) :

```
FD  TIME-CARD-FILE
    BLOCK CONTAINS 0 RECORDS
    RECORD CONTAINS 33 CHARACTERS
    LABEL RECORDS ARE STANDARD

01  TIME-CARD-RECORD                      PIC X(33).

FD  TIME-CARD-EDIT-REPORT
    RECORD CONTAINS 132 CHARACTERS
    LABEL RECORDS ARE OMITTED
    .
01  TIME-CARD-EDIT-LINE                   PIC X(132).

WORKING-STORAGE SECTION.

01  WS-TIME-CARD-REC.
    05  WS-TIME-CARD-ID                   PIC X(6).
    05  WS-TIME-CARD-NAME                 PIC X(20).
    05  WS-TIME-CARD-DEPARTMENT           PIC X(4).
    05  WS-TIME-CARD-HOURS                PIC S99V9.

01  WS-TIME-EDIT-LINE.
    05  WS-TIME-EDIT-ID                   PIC X(6).
    05  FILLER                            PIC X(4)    VALUE SPACES.
    05  WS-TIME-EDIT-NAME                 PIC X(20).
    05  FILLER                            PIC X(10)   VALUE SPACES.
    05  WS-TIME-EDIT-DEPARTMENT           PIC X(4).
    05  FILLER                            PIC X(6)    VALUE SPACES.
    05  WS-TIME-EDIT-HOURS                PIC ZZ.9--.
    05  FILLER                            PIC X(77)   VALUE SPACES.

01  NUMBER-TO-SKIP                        PIC S9(3)   COMP-3.

PROCEDURE DIVISION.

    OPEN    INPUT     TIME-CARD-FILE
            OUTPUT    TIME-CARD-EDIT-REPORT

    READ TIME-CARD-FILE
        INTO WS-TIME-CARD-REC
        AT END . . .
    .
    MOVE WS-TIME-CARD-ID               TO WS-TIME-EDIT-ID
    MOVE WS-TIME-CARD-NAME             TO WS-TIME-EDIT-NAME
    MOVE WS-TIME-CARD-DEPARTMENT       TO WS-TIME-EDIT-DEPARTMENT
    MOVE WS-TIME-CARD-HOURS            TO WS-TIME-EDIT-HOURS

    WRITE TIME-CARD-EDIT-LINE FROM WS-TIME-EDIT-LINE
        BEFORE ADVANCING 3 LINES
```

يتسبب WRITE... BEFORE ADVANCING 3 LINES فى طباعة المحتويات الحالية لـ TIME- EDIT- LINE عن طريق TIME- CARD- EDIT- LINE ، وذلك عندما يكون الورق موجودا فى الطابع . بعد طباعة السطر .. يتقدم الورق ثلاثة سطور للامام . وعلى هذا إذا نفذت WRITE مرات ومرات .. فتكون النتيجة ترك سطرين فارغين بين السطور المطبوعة .

يستغل المثال منهج مخزن العمل ، من حيث استخدام جزء VALUE فى وضع مناطق FILLER ( التى تفصل الحقول على الورق المطبوع ) على أنها فراغات . فإذا بنينا TIME- EDIT- LINE فى قسم الملفات ( حيث لايسمح باستخدام VALUE ) .. فيجب أن نستخدم عبارة MOVE فى نقل فراغات SPACES الى منطقة السجل المنطقى قبل الانتقال إلى المعلومات المراد طباعتها .

إذا لم يوجد الجزء BEFORE ADVANCING 3 LINES فى البرنامج السابق ، يعمل الجزء التلقليدى AFTER ADVANCING 1 LINE . وعلى هذا.. لاتوجد سطور فارغة بين السطور المطبوعة .

**مثـال ٦ - ١٣ :**

لتوضيح استخدام هذا الجزء

```
WRITE . . . BEFORE/AFTER ADVANCING identifier-2 LINES
```

افرض - كمثال - أننا عرفنا اسم بيانات NUMBRE- TO - SKIP ، والذى يجب أن يكون له PICTURE عددى بدون علامة عشرية . عند ذلك :

```
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
MOVE 3 TO NUMBER-TO-SKIP
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
WRITE TIME-CARD-EDIT-LINE
    FROM WS-TIME-EDIT-LINE
    AFTER ADVANCING NUMBER-TO-SKIP LINES
```

يمكن أن تكون محتويات NUMBRE- TO - SKIP أى مدخلات صحيحة اختيارية يدخلها البرنامج، أو تنتج من حسابات مبرمجة . يتسبب WRITE... AFTER ADVANCING فى تقويم الورق بعدد السطور المحدد ، ثم إخراج السطر المراد طباعته .

**مثـال ٦ - ١٤ :**

ينقل بديل BEFORE/AFTER ADVANCING PAGE الورق إلى بداية الصفحة التالية، وإذا [illegible] جزء الخطية LINAGE فى FD للملف ، الورق يوضع عند بداية جسم الصفحة للصفحة المنطقية التالية (انظر القسم رقم 7 من الفصل الخامس). وإلا.. تم وضع الورق عند القناة 1 لآلية التحكم فى عربة الطابع ( والتى تناظر بصفة عامة السطر 1 فى الصفحة التالية، انظر القسم الرابع من الفصل الرابع ) .

**مثـال ٦ - ١٥ :**

يسمح جزء BEFORE/AFTER ADVANCING mnemonic- name بوضع الورق فى قناة تحكم العربة المعرفة، فى مقطع الأسماء الخاصة من جزء الأوساط (راجع قسم ٤ - ٤) .

```
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SPECIAL-NAMES.
    C01 IS TOP-OF-FORM
    C02 IS HEADING-AREA
    C05 IS TOTAL-AREA
    .
.....................
PROCEDURE DIVISION.
    WRITE REPORT-LINE
        FROM WS-PAGE-NUMBER-DATE-LINE
        AFTER ADVANCING TOP-OF-FORM
    WRITE REPORT-LINE
        FROM WS-HEADING-LINE
        AFTER ADVANCING HEADING-AREA
    WRITE REPORT-LINE
        FROM WS-TOTAL-LINE
        AFTER ADVANCING TOTAL-AREA
```

هنا .. يقدم AFTER ADVANCING TOP- OF- FORM الورق إلى قناة تحكم العربة رقم 1 ، ثم تطبع المحتويات الحالية لـ WS- PAGE- NUMBER- DATE- LINE .. إلخ. وقبل السماح بتنفيذ البرنامج ، يجب أن يعد مشغل الكمبيوتر الطابع بحيث تكون قنوات التحكم فى العربة المستخدمة - بواسطة البرنامج - متفقة مع الأجزاء المناسبة من الصيغ المستمرة ؛ فإذا لم يحدث ذلك .. فلن تكون مسافات الطباعة مضبوطة .

إن استخدام قنوات التحكم فى العربة المعرفة فى مقطع الاسماء الخاصة واستخدام جزء الخطية LINAGE فى FD مستقلان عن بعضهما البعض .

فى شكل (٦ - ٣) "AT END- OF- PAGE" و "AT EOP" متكافئان . ويمكن استخدامهما عندما يكون جزء LINAGE محددا فى FD فقط .. فى هذه الحالة ، يتسبب مترجم الكوبل فى حفظ برنامج الهدف تلقائيا لصيانة LINAGE- COUNTER الذى يعد كل السطور المطبوعة أو المتروكة فى صفحة منطقية . وكلما بدأت صفحة منطقية جديدة .. يعاد وضع القيمة 1 فى LINAGE- COUNTER بصورة تلقائية .

تنفذ العبارات المحددة فى جزء AT END- OF- PAGE عندما يتسبب تنفيذ WRITE BEFORE/AFTER ADVANCING فى أن يتعدى LINAGE- COUNTER ( ١ ) القيمة المحددة فى ...WITH FOOTING AT (يختلف حجم جسم الصفحة عما هو محدد بواسطة LINAGE IS) أو ( ٢ ) حجم جسم الصفحة .

- **الحالة الأولى :** تنفذ عبارة WRITE كما هو الحال المعتاد، وتنفذ العبارات فى جزء AT END- OF- PAGE، ثم يستأنف التنفيذ المعتاد للعبارات .
- **الحالة الثانية :** إذا تحدد BEFORE ADVANCING .. يطبع السطر ، ثم يتقدم الورق إلى بداية جسم الصفحة المنطقية التالية ، وإذا تحدد AFTER ADVANCING .. يتقدم الورق إلى بداية جسم الصفحة المنطقية التالية ، ثم يطبع السطر، وتنفذ بعد ذلك عبارات AT END- OF- PAGE .

**مثـال ٦ – ١٦ :**

تسمى الحالة الثانية بالسريان الزائد التلقائى للصفحة automatic page overflow . افرض انه بالنسبة إلى RE- PORT- FILE :

```
LINAGE IS 60
    WITH FOOTING AT 53
    LINES AT TOP 3
    LINES AT BOTTOM 3
```

وأن الورق موجود عند السطر الثانى والخمسين داخل جسم الصفحة حاليا . ويصمم تنفيذ مايلى :

```
WRITE REPORT-LINE
    FROM WS-CHECK-REQUEST-EDIT-LINE
    AFTER ADVANCING 10 LINES
    AT END-OF-PAGE
        MOVE WS-RUNNING-TOTAL TO WS-OUTPUT-RUNNING-TOTAL
        WRITE REPORT-LINE FROM WS-RUNNING-TOTAL-LINE
            AFTER ADVANCING 5 LINES
```

لطباعة إجمالى جار running total فى منطقة النهاية ، فى نهاية جسم الصفحة . إلا أنه حيث إن AFTER ADVANCIING 10 LINES يتسبب فى سريان زائد للصفحة (١٠ + ٥٢ = ٦٢) ، والتى تتعدى حجم جسم الصفحة التى لها (LINAGE 60) فيحدث مايلى :

( ١ ) يتقدم الورق إلى أول سطر طباعة فى الصفحة المنطقية التالية .

( ٢ ) تطبع محتويات WS- CHECK- REQUEST- EDIT- LINE .

( ٣ ) نظراً لحدوث سريان زائد تلقائى للصفحة.. فإن جزء AT END- OF- PAGE يتسبب فى تقديم الورقة خمسة أسطر أخرى، وتطبع بعدها محتويات WS- RUNNING- TOTAL- LINE .

وعلى هذا.. يظهر سطر الإجمالى الجارى بالقرب من بداية جسم الصفحة ، مختلطًا مع أسطر التقرير المعتادة – كوضع غير مرغوب فيه .

**مثـال ٦ – ١٧ :**

يمكن أن يحذف المبرمج صعوبات أجزاء AT END- OF- PAGE ، وذلك بالتنسيق المحتمل لقيمة BEFORE/AFTER ADVANCING ، مع قيمة WITH FOOTING AT ، وجزء AT END- OF- PAGE .

اجعل جزء LINAGE كما هو فى مثال (٦ – ١٦) . افرض كذلك أن فى بداية كل جسم صفحة يطبع عنوانًا يترك الورق موضوعا عند السطر رقم 10 من جسم الصفحة . أخيرا ، افرض أنه مطلوب طباعة سطور تقرير، مع وجود سطرين فارغين بين السطور المطبوعة . أعتبر WRITE التالية :

```
WRITE REPORT-LINE
    FROM WS-CHECK-REQUEST-EDIT-LINE
    AFTER ADVANCING 3 LINES
    AT END-OF-PAGE
        MOVE WS-RUNNING-TOTAL TO WS-OUTPUT-RUNNING-TOTAL
        WRITE REPORT-LINE
            FROM WS-RUNNING-TOTAL-LINE
            AFTER ADVANCING 5 LINES
```

```
WRITE REPORT-LINE
    FROM WS-FIRST-TITLE-LINE
    AFTER ADVANCING PAGE
WRITE REPORT-LINE
    FROM WS-SECOND-TITLE-LINE
    AFTER ADVANCING 9 LINES
```

عندما تنفذ عبارة WRITE هذه لاول مرة فإنها تجعل الورق موضوعا عند نهاية العنوان ( السطر العاشر) . وعلى هذا.. يصبح أول سطر فى التقرير الثالث عشر (AFTER ADVANCING 3 LINES) ، ويصبح السطر الثانى فى السطر السادس عشر... ويقع السطر الرابع عشر من التقرير فى السطر ٥٢ . يؤدى التنفيذ الخامس عشر لعبارة WRITE إلى حدوث مايلى :

( ١ ) طباعة السطر الخامس عشر من التقرير فى السطر رقم ٥٥ من الصفحة .

( ٢ ) حيث إن LINAGE- COUNTER ( ٥٥ الآن ) تعدى حد منطقة النهاية ( WITH FOOTING AT 53)... يتسبب جزء AT END- OF- PAGE فى تقديم الورقة الى السطر رقم ٦٠ . (AFTER ADVANCING 5 LINES) ، وتطبع محتويات WS- RUNNING- TOTAL- LINE على السطر رقم ٦٠ ، ( آخر سطر فى جسم الصفحة ) ، كما يكتب أول سطر من عنوان الصفحة التالية على السطر رقم 1 من جسم الصفحة التالية (AFTER ADVANCING PAGE) ، وأخيرا ، تكتب بقية العنوان على السطر العاشر (AFTER ADVANCING 9 LINES) من السطر رقم 1 ) .

لاحظ عدم قيام جزء AT END- OF- PAGE بطبع الإجمالى الجارى فى منطقة النهاية من كل صفحة فقط ، لكنه يقدم الورق إلى الصفحة المنطقية التالية ويطبع العنوان . يترك الورق موضوعا عند السطر العاشر، بحيث تبدأ الدورة مرة أخرى .

مثل AT END ( القسم الرابع من هذا الفصل ) فيفترض أن تستمر AT END- OF- PAGE حتى تظهر نقطة .

**مثـال ٦ – ١٨ :**

```
READ CHECK-REQUEST-FILE
    AT END
        MOVE "T" TO CHECK-REQUEST-END-SW
MOVE CHECK-REQUEST-VENDOR-ID  TO  WS-VENDOR-ID
MOVE CHECK-REQUEST-AMOUNT     TO  WS-AMOUNT
WRITE CHECK-REQUEST-EDIT-LINE
    FROM WS-EDIT-LINE
    AFTER ADVANCING 2 LINES
    AT END-OF-PAGE
        WRITE CHECK-REQUEST-EDIT-LINE
            FROM CHECK-REQUEST-EDIT-TITLE
            AFTER ADVANCING PAGE
   D 1 TO NUMBER-CHECK-REQUESTS
```

يتطلب جزء البرنامج هذا نقطة مرتبة بعد " MOVE " T ، ونقطة مرتبة أخرى بعد AFTER ADVANCING PAGE. كما تعنى.. فإن الترحيل لايعنى شيئًا، يستمر جزء AT END حتى نهاية الجزء . وعلى هذا .. فإنه يشمل أول عبارة WRITE ، هى عبارة شرطية ؛ لأنها تحتوى على جزء AT END- OF- PAGE ، الذى ينفذ تحت شروط معينة . وعلى هذا يشملAT END خطأ تكوينيًا ؛ لانه يجب ألا يحتوى إلا على عبارات أمرية غير شرطية ( انظر القسم الرابع من هذا الفصل السادس ) .

## ٦ - ٦ مدخلات : عبارة القبول

تستخدم عبارة القبول ACCEPT لمدخلات مخصصة معينة .

## قبول معلومات من نظام التشغيل

التكوين لنقل معلومات نظام تشغيل معينة إلى برنامج هدف الكوبل، هو :

```
                             { DATE }
ACCEPT identifier FROM       { DAY  }
                             { TIME }
```

حيث identifier هو اسم بيانات، معرف بطريقة مناسبة فى جزء البيانات .

**مثال ٦ - ١٩ :**

```
•   WORKING-STORAGE SECTION.
    01  SAMPLE-ACCEPTED-DATE      PIC 9(6).

    PROCEDURE DIVISION.
        ACCEPT SAMPLE-ACCEPTED-DATE FROM DATE
```

يضع نظام التشغيل التاريخ الحالى فى عنصر البيانات SAMPLE- ACCEPTED- DATE. والصورة المناسبة للتاريخ DATE هى PIC9 (6). وتكون صورة التاريخ yymmdd، حيث yy هى السنة، وmm هو الشهر، وdd هو اليوم، أى ان ١٢ فبراير تمثل على النحو التالى 830212.

```
•   01  SAMPLE-ACCEPTED-DAY      PIC 9(5).
        ACCEPT SAMPLE-ACCEPTED-DAY FROM DAY
```

يضع نظام التشغيل التاريخ الميلادى اليومى julian date فى عنصر البيانات SAMPLE- ACCEPTED- DAY والصورة المناسبة لليوم DAY ، هى PIC 9 (5) . ويمثل التاريخ الميلادى على هيئة عدد من خمسة أرقام فى الصورة yyddd؛ حيث yy هى السنة ddd هو اليوم ( حيث ترقم الأيام من ١ إلى ٣٦٥، أو من ١ إلى ٣٦٦ فى السنة الكبيسة ) . ويمثل ١٢ فبراير ١٩٨٣ على النحو التالى 83043 :

```
●    01 SAMPLE-ACCEPTED-TIME     PIC 9(8).
        ACCEPT SAMPLE-ACCEPTED-TIME  FROM  TIME
```

يضع نظام التشغيل الوقت من اليوم فى عنصر البيانات SAMPLE- ACCEPTED- TIME ، والصورة المناسبة للوقت TIME هى PIC 9 (8). ويمثل الوقت من اليوم على هيئة عدد من ثمانية أرقام فى الصورة HHmmsshh ، حيث HH : هى الساعة ( من ٠ إلى ٢٣ ) ، وMM هى الدقائق ( من ٠ إلى ٥٩ ) ، SS هى الثوانى ( من ٠ إلى ٥٩ ) ، وhh هى جزء من المائة من الثانية ( من ٠ إلى ٩٩ ) . وعلى هذا.. تظهر الساعة 24 : 3 بعد الظهر فى الصورة التالية 15240000 .

## قبول بيانات من SYSIN، أو من نهاية طرفية يعمل عليها مشغل الكمبيوتر

يمكن استخدام الصيغة الأخرى لعبارة ACCEPT فى إدخال كم بسيط من المعلومات من ملف نظام خاص أو ، الأكثر اعتيادا ، من نهاية طرفية يعمل عليها مشغل الكمبيوتر . والتكوين هو كما يلى :

<u>ACCEPT</u> identifier [<u>FROM</u> special-name]

يجب أن يعرف special- name فى مقطع الأسماء الخاصة من جزء الأوساط ( القسم الرابع من الفصل الرابع). وتختلف التفاصيل الدقيقة لصيغة ACCEPT هذه من نظام لآخر ، الا ان المحتويين المقبولين فى قسم الأسماء الخاصة فى كوبل IBM OS/VS لعبارة ACCEPT ، هما : CONSOLE و SYSIN :

```
SPECIAL-NAMES.
    CONSOLE  IS  OPERATOR-MESSAGE-DEVICE
    SYSIN        IS  BATCH-CONTROL-CARD
```

كلمة CONSOLE .. هى إحدى كلمات الكوبل المحجوزة من كوبل IBM ، والتى تحدد النهاية الطرفية التى يعمل عليها مشغل الكمبيوتر، التى تستخدم فى الاتصال بنظام التشغيل وبرامج التنفيذ الأخرى من خلال نظام التشغيل . كلمة SYSIN هى كلمة محجوزة فى نظم IBM ، تحدد ملف خاص له سجلات اطوالها ٨٠ بايت؛ فإذا أراد المبرمج استخدام ملف SYSIN .. فيجب تقديم عبارة لغة تحكم عمل لتعريفة . وبصفة عامة.. يجب تجنب استخدام ACCEPT فى الإدخال من ملف SYSIN الا إذا كانت المعلومات المشمولة صغيرة جدا . من الأفضل إعداد ملف معتاد بعبارة SELECT ، واستخدام FD و OPEN و READ .

**مثـال (٦ - ٢٠) :**

```
SPECIAL-NAMES.
    CONSOLE  IS  OPERATOR-MESSAGE-DEVICE
```

```
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
WORKING-STORAGE SECTION.
01  NUMBER-OF-LABELS-ACROSS    PIC 9    DISPLAY.
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
PROCEDURE DIVISION.
    ACCEPT NUMBER-OF-LABELS-ACROSS FROM OPERATOR-MESSAGE-DEVICE
```

عندما تطبع العناوين البريدية أو الشيكات أو أى صيغ صغيرة أخرى باستخدام الكمبيوتر ، فيمكن زيادة سرعة الطباعة بطباعة العديد من الصيغ فى السطر الواحد . ويوضح الجزء السابق من برنامج كيف يتمكن مشغل الكمبيوتر من تحديد عدد العناوين التى يجب أن تطبع فى الصفحة فى هذا التنفيذ الخاص للبرنامج . عندما تنفذ عبارة ACCEPT ، يتوقف تنفيذ بقية البرنامج مؤقتا حتى يكتب مشغل الكمبيوتر المعلومات المطلوبة مستخدما لوحة مفاتيح النهاية الطرفية . عند ذلك توضع المعلومات فى عنصر البيانات المحدد بواسطة عبارة ACCEPT ، وهـو NUMBER- OF- LABELS- ACROSS . لأن البيانات المقبولة تأتى من نهاية طرفية ملحق بها شاشة، أو ملحق بها طابع ، فيفضل استخدام DISPLAY . يجب أن يكتب مشغل الكمبيوتر المعلومات طبقا للصورة PICTURE الموجودة فى برنامج الكوبل ، وإلا فسوف تحدث أخطاء .

**مثـال ٦ - ٢١ :**

من ضمن الاستخدامات القليلة العملية لعبارة ACCEPT من SYSIN ، هى تقديم مدخلات بطاقة تحكم . بطاقات التحكم هى control cards هى سجلات ( عادة ماتكون مثقبة فى بطاقات ) تحدد بدائل البرنامج التى تعمل بالنسبة لتشغيل الكمبيوتر الحالى . افرض أن برنامج العناوين البريدية يرتب العناوين؛ طبقا للرمز البريدى أو طبقا للاسم ، وأنه يمكن أن يشتمل أو لا يشتمل على العاملين بالشركة .

```
SPECIAL-NAMES.
    SYSIN IS OPTION-INPUT-DEVICE
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
WORKING-STORAGE SECTION.
01  OPTIONS-FOR-THIS-RUN.
    05  TYPE-OF-SORT          PIC  X(4)
    05  INCLUDE-EMPLOYEES     PIC  X(3).
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
PROCEDURE DIVISION.
    ACCEPT OPTIONS-FOR-THIS-RUN FROM OPTION-INPUT-DEVICE
```

يمكن أن تحتوى بطاقة التحكم على مايلى :

```
NAMENOb        or        ZIPbYES
X(4) X(3)                X(4) X(3)
```

لتتسبب فى أن البرنامج يرتب العناوين طبقا للاسم مع حذف العاملين من البريد ، أو للترتيب طبقا للرقم البريدى مع شمول العاملين وذلك على التوالى . يدخل كل تنفيذ لعبارة ACCEPT سجل منطقى جديد من ملف SYSIN . لاحظ أنه بالرغم من أن طول سجلات SYSIN هو ٨٠ بايت ، إلا أنه ليس هناك حاجة لحساب بايت إضافى .

73 = (4+3) - 80 فى وصف OPTIONS- FOR- THIS- RUN .

## ٦ - ٧ مخرجات : عبارة العرض

تصمم عبارة العرض DISPLAY لانتاج مخرجات مطبوعة محدودة الحجم على ملفات نظام خاصة أو على النهاية الطرفية التى يستخدمها مشغل الكمبيوتر . ومثل نظيرتها ACCEPT .. فإن التفاصيل الدقيقة لعبارة العرض تتغير من نظام لآخر وتكوينها فى كوبل IBM OS/VS هو كما يلى :

$$\underline{\text{DISPLAY}} \begin{Bmatrix} \text{identifier-1} \\ \text{literal-1} \end{Bmatrix} \begin{bmatrix} \text{identifier-2} \\ \text{literal-2} \end{bmatrix} \ldots [\underline{\text{UPON}}\ \text{special-name}]$$

حيث يجب تعريف special- name فى مقطع الأسماء الخاصة ( القسم الرابع من الفصل الرابع ) . والمحتويات المقبولة فى مقطع الأسماء الخاصة لعبارة DISPLAY هى :

```
SPECIAL-NAMES.
    CONSOLE IS...
    SYSOUT IS...
```

وهنا CONSOLE هى نفسها كما ذكرت فى القسم السادس من هذا الفصل، و SYSOUT هى المخرجات المناظرة لـ SYSIN.

**مثـال ٦ - ٢٢ :**

يمكن استخدام DISPLAY ( بالاتصال مع ACCEPT ) فى التداخل مع مشغل الكمبيوتر، وذلك عن طريق النهاية الطرفية لمشغل الكمبيوتر .

```
SPECIAL-NAMES.
    CONSOLE IS OPERATOR-MESSAGE-DEVICE
WORKING-STORAGE SECTION.

01  WS-OPERATOR-REPLIES.
    05  WS-NUMBER-OF-LABELS-ACROSS     PIC 9.
    05  WS-EMPLOYEES-INCLUDED          PIC X(3).

PROCEDURE DIVISION.

    DISPLAY "PLEASE SET UP PRINTER FOR MAILING LABELS"
        UPON OPERATOR-MESSAGE-DEVICE

    DISPLAY "WHEN READY, ENTER NUMBER OF LABELS ACROSS (1 DIGIT)"
        UPON OPERATOR-MESSAGE-DEVICE

    ACCEPT WS-NUMBER-OF-LABELS-ACROSS
        FROM OPERATOR-MESSAGE-DEVICE

    DISPLAY "EMPLOYEES INCLUDED? ('YES' OR 'NO')"
        UPON OPERATOR-MESSAGE-DEVICE

    ACCEPT WS-EMPLOYEES-INCLUDED
        FROM OPERATOR-MESSAGE-DEVICE
```

الملقنات prompts المطبوعة على النهاية الطرفية المتاحة لمشغل الكمبيوتر توجه المشغل لتنفيذ أنشطة معينة أو ترشده لإدخال معلومات لعبارة ACCEPT .

ويجب أن تتذكر أنه عندما يعرض DISPLAY ثابتًا غير عددى ( مثل الملقنات سالفة الذكر ) على وحدة مخرجات تشتمل على سطور.. طول السطر 100 رمز مثلا ؛ فتتجزأ المخرجات إلى وحدات من 100 رمز ، مع حدوث التجزئة فى أى موقع بعد المائة رمز . وفى حالة الثوابت الاستعارية.. يطبع رمز واحد ( المخرجات من DISPLAY SPACES هى فراغ واحد فقط ) . وتحول DASPLAY كلاً من COMP، أو COMP-3 بصورة تلقائية إلى DISPLAY قبل الطباعة . وفى كوبل IBM OS/VS ، تحذف إشارات الموجب تلقائيا، أما إشارات السالب فلا تحذف (انظر جدول ٥ – ١) لاتطبع العلامة العشرية المعرفة بصورة V.

**مثــال ٦ – ٢٣ :**

استخدام آخر اساس لعبارة DISPLAY هو فى أسطر التصحيح (انظر القسم الثاني من الفصل الرابع).

```
       SPECIAL-NAMES.
D          SYSOUT IS DEBUGGING-DEVICE

       PROCEDURE DIVISION.
D          DISPLAY "STARTING VALUE OF WS-NUMBER-OF-CLIENTS IS "
D              WS-NUMBER-OF-CLIENTS
D              UPON DEBUGGING-DEVICE
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
D          DISPLAY "WHEN ID INVALID, EMPLOYEE NAME IS "
D              EMPLOYEE-MASTER-NAME
D              UPON DEBUGGING-DEVICE
```

عندما يحدد مقطع كمبيوتر المصدر WITH DEBUGGING MODE، تترجم اسطر التصحيح (التى لها الحرف D فى العمود السابع) إلى برنامج هدف متسببة فى معلومات مفيدة عن محتويات عناصر البيانات التى تطبع فى ملف SYSOUT. لاحظ أن الثوابت غير العددية تشمل فراغا اضافيا (قبل علامة التنصيص التى تنهيها) لفصل العناصر فى سطر DISPLAY، ولاتقدم DASPLAY فراغات فاصلة بصورة تلقائية.

**مثــال ٦ – ٢٤ :**

الاستخدام الثالث الرئيسى لعبارة DISPLAY هو فى طباعة رسائل خطأ أو رسائل استثناءات على ملف طباعة SYSOUT، بحيث أنه يمكن شد انتباه الشخص إلى شروط الخطأ والاستثناءات المكتشفة أثناء تنفيذ البرنامج. (يجب الا يستخدم هذا الاستخدام عندما يكون متوقعا حدوث عدد معنوى للأخطاء وللاستثناءات. وعلى هذا، فعادة مايتم تشغيل المدخلات المكتوبة بلوحة المفاتيح بواسطة برنامج تنقيح edit program، والذى تكون وظيفته الرئيسية اكتشاف الأخطاء فى بيانات المدخلات وطباعة تقرير خطأ معطيا معلومات كافية بحيث يمكن تصحيح الاخطاء. فى هذه الحالات، يجب أن تطبع معلومات الاخطاء فى ملف معتاد، والذى يعد بواسطة SELECT , FD.. إلخ).

```
SPECIAL-NAMES.
    SYSOUT IS ERROR-MESSAGE-LOG
. . . . . . . . . . . . . . . . . . . . . . . . .
PROCEDURE DIVISION.
. . . . . . . . . . . . . . . . . . . . . . . . . . . .

OPEN  INPUT  CHECK-REQUEST-FILE
IF CHECK-REQUEST-STATUS-CODE NOT EQUAL "00"
    DISPLAY "UNABLE TO OPEN CHECK REQUEST FILE"
        UPON ERROR-MESSAGE-LOG
```

إذا لم يستخدم جزء UPON فى عبارة DISPLAY فالإجراء التقليدى في كوبل IBM OS/VS هو:

```
DISPLAY . . . UPON SYSOUT
```

## ٦ - ٨ تشغيل : عبارة النقل

فعل النقل MOVE هو فعل مضلل، فتأثير النقل هو نسخ copy محتويات احد عناصر البيانات في عنصر بيانات آخر أو في عناصر بيانات أخرى. لايتغير الحقل الراسل sending field عند تنفيذ عبارة النقل، وتستبدل المحتويات الاصلية فى الحقل المستقبل recieving field بمحتويات الحقل الراسل. تتسبب عبارة النقل كذلك فى تحويل البيانات من حالة استخدام USAGE الحقل المستقبل، إذا ما اختلفت الحالتان. أكثر من هذا، إذا اختلف طول الحقل الراسل عن طول الحقل المستقبل، فتتسبب عبارة النقل فى الحذف أو فى ملأ فراغات أو أصفار فى الحقل المستقبل. أخيرا، تتسبب عبارة النقل فى تنقيح edit البيانات العددية إذا كانت صورة PICTURE الحقل المستقبل محتوية على رموز تنقيح.

وعبارة النقل لها التكوين التالى:

MOVE {identifier-1 | literal} TO identifier-2 [identifier-3] . . .

( وهناك تكوين بديل لن يعالج فى هذا الكتاب ، الا انه مذكور فى مثال ٦ - ٣١ ) .

مثـال ٦ - ٢٥ :

بمعرفة مايلى :

```
01  SAMPLE-MOVE-ITEMS.
    05  A      PIC  X(5).
    05  B      PIC  X(5).
    05  C      PIC  X(3).
    05  D      PIC  X(7).
```

افرض أن الحقل A يحتوى على "HELLO" . وعلى هذا.. فإن :

- MOVE A TO B

تنسخ محتويات عنصر البيانات A (identifier-1) فى عنصر البيانات B (iden tifier-2) . وحيث أن A و B لهما نفس الصورة والاستخدام.. فأن كلاً منهما يحتوى على HELLO بعد تنفيذ عبارة النقل . وتفقد المحتويات الأصلية للحقل B .

- MOVE A TO C

تنسخ محتويات العنصر A فى العنصر C، وحيث أن صورة العنصر C هى PIC X(3) يحذف جزء من HELLO من ناحية اليمين : بعد تنفيذ عبارة النقل.. يحتوى A على HELLO ، بينما يحتوى C على HEL .

- MOVE A TO D

تنسخ محتويات العنصر A فى العنصر D ، حيث تملأ صورة العنصر D هى PIC X(7) المواقع الأكثر بفراغات من ناحية اليمين : بعد تنفيذ عبارة النقل.. يحتوى A على HELLO، بينما يحتوى D على HELLObb .

- MOVE A TO B C D

تنسخ محتويات عنصر البيانات A فى عناصر البيانات : B و C و D وينتج عنها نفس لغة الآلة مثل عبارات النقل الثالث السابقة تماما .

- MOVE SPACES TO A

تنسخ خمس فراغات ( هنا literal هو الثابت الاستعارى SPACES ) فى عنصر البيانات A(identifier-2)، والمعرّف بصورة PIC X(5) .

- MOVE ZEROS TO A C

تضع 00000 فى A، وتضع 000 فى C .

- MOVE ALL "-" TO C D

تضع --- فى C وتضع ------- فى D .

لاحظ مرة أخرى.. استخدام الثابت الاستعارى " - " ALL ، الذى يقدم العدد الصحيح والنوع الصحيح للرموز تلقائيا

**مثـال ٦ - ٢٦ :**

```
01  SAMPLE-NUMERIC-ITEMS.
    05  A     PIC  S9(5)     VALUE  +12345.
    05  B     PIC  S9(5).
    05  C     PIC  S9(3).
    05  D     PIC  S9(7).
```

- MOVE A TO B

حيث إن كلا من A و B له صورة PIC 9(5)، وبسبب استخدام DISPLAY ، فإنهما يحتويان نفس القيمة بعد تنفيذ عبارة النقل : 12345+ .

- MOVE A TO C

يحدث الحذف من ناحية اليسار بالنسبة الى العناصر العددية، يحتوى C على 345+ [ حيث أن صورته هى S9 (3)] ، ويظل A كما هو دون تغيير .

- MOVE A TO D

تضاف الأصفار الزائدة للعناصر العددية من ناحية اليسار، ويظل A كما هو دون تغيير ، بينما يحتوى D على 0012345+ .

**مثـال ٦ – ٢٧ :**

بمعرفة مايلى :

```
01  SAMPLE-NUMERIC-ITEMS.
    05  A   PIC  S9(3)V99    COMP-3     VALUE  +123.45.
    05  B   PIC  S9(3)V99    DISPLAY.
    05  C   PIC  S9(3)V99    COMP.
    05  D   PIC  S9(5)V99    COMP-3.
    05  E   PIC  S9(5)V9     DISPLAY.
    05  F   PIC  S9(5)V999   COMP.
    05  G   PIC  S9(2)V9     COMP-3.
```

- MOVE A TO B

A و B لهما نفس الصورة ، ولهما استخدام USAGE مختلف . وتتحول قيمة COMP-3 فى A الى DISPLAY، وتنسخ فى B . وعلى هذا ، يحتوى كل من A و B على 123.45+ بعد النقل ، إلا أن قيمة A تخزن فى صورة COMP-3، أما قيمة B فتخزن فى صورة DISPLAY .

- MOVE A TO D

الاستخدام هنا هو نفسه ، ولكن الصورة مختلفة . تضبط العلامات العشرية ، ثم توضع أصفار أضافية فى الحقل المستقبل، وذلك من ناحية اليسار . يظل A كما هو دون تغيير ، ويحتوى D على 00123.45+ فى صورة COMP-3 . تحفظ العلامات العشرية فى الحقول الراسلة والمستقبلة مضبوطة دائما .

- MOVE A TO E

يحدث تحويل تلقائى من COMP-3 الى DISPLAY . بالإضافة إلى ذلك ، تضاف أصفار إلى الجزء الصحيح من العدد (من ناحية اليسار) .

وحيث إن E لها موقع لكسر عشرى واحد ، E تحتوى على 00123.4+ والاستخدام DISPLAY بعد النقل . لاحظ أنه لم يحدث تقريب rounding بواسطة عبارة النقل ، وأن العلامة العشرية ظلت فى نفس موقعها .

- MOVE A TO F

يحدث هنا تحويل تلقائى من COMP-3 الى COMP . أضيفت اصفار إلى الرقم الصحيح من العدد ( من ناحية اليسار). حيث أن الحقل المستقبل يحدد مواقع عشرية أكثر من الحقل الراسل.. تضاف أصفار كذلك فى الكسر العشرى ( من ناحية اليمين ) . يظل A كما هو دون تغيير بعد النقل و يحتوى F على 00123.450+ والاستخدام COMP .

- MOVE A TO G

كلا الاستخدامين من نوع COMP-3 ، بحيث أنه لاينفذ اى تحويل، ونظراً لأن صورة G هى PIC S 9(2)V9 ، يحدث حذف من ناحية اليمين وناحية اليسار . وتكون نتيجة G هى 23.4+ من نوع COMP-3 .

- MOVE A TO B D E F G

ينتج عن هذه العبارة نفس لغة الآلة تماما، مثل العبارات الخمس السابق ذكرها .

- MOVE ZEROS TO A B C

حيث إن كلاً من A , B , C له نفس الصورة.. فإن كلا منها يحتوى على 000.00+. مع ملاحظة أن كل حقل يحتوى على نوع صفرZERO مختلف، كمايلى: A يكون من نوع B, COMP-3 من نوع DISPLAY , C من نوع COMP. وعندما تستخدم الثوابت الاستعارية فى عبارة النقل أو فى جزء VALUE ، فانها تمثل الكمية الصحيحة correct amount من البيانات، والاستخدام USAGE الصحيح دائما .

**مثـال ٦ - ٢٨ :**

بمعرفة مايلى :

```
01  SAMPLE-NUMERIC-ITEMS.
    05  A          PIC S9(5)V99      VALUE +00010.08.
    05  A-EDITED   PIC $$$,$$$.99-.
MOVE A TO A-EDITED
```

عندما ينقل عنصر عددى إلى حقل به رموز تنقيح.. يشمل جزء من تنفيذ عبارة النقل تنقيحاً للقيمة فى الحقل المستقبل . وتظل القيمة فى A كما هى دون تغيير ، أما القيمة فى A-EDITED بعد النقل فهى : bbbb$10.058b

**مثال ٦ - ٢٩ :**

( عبارات نقل غير صحيحة ) : يصنف الكوبل عناصر البيانات فى ست فئات بيانات categories of data و باعتبار literal-1 و literal-2 فئتين من هذه الفئات ، تكون لدينا القائمة التالية بعدد ١٤ عبارة نقل صحيحة .

- MOVE ALPHABETIC-DATA TO NUMERIC-INTEGER-DATA
- MOVE ALPHABETIC-DATA TO NUMERIC-NONINTEGER-DATA
- MOVE ALPHABETIC-DATA TO NUMERIC-EDITED-DATA
- MOVE ALPHANUMERIC-EDITED-DATA TO NUMERIC-INTEGER-DATA
- MOVE ALPHANUMERIC-EDITED-DATA TO NUMERIC-NONINTEGER-DATA
- MOVE ALPHANUMERIC-EDITED-DATA TO NUMERIC-EDITED-DATA
- MOVE NUMERIC-INTEGER-DATA TO ALPHABETIC-DATA
- MOVE NUMERIC-NONINTEGER-DATA TO ALPHABETIC-DATA
- MOVE NUMERIC-NONINTEGER-DATA TO ALPHANUMERIC-DATA
- MOVE NUMERIC-NONINTEGER-DATA TO ALPHANUMERIC-EDITED-DATA
- MOVE NUMERIC-EDITED-DATA TO ALPHABETIC-DATA
- MOVE NUMERIC-EDITED-DATA TO NUMERIC-INTEGER-DATA
- MOVE NUMERIC-EDITED-DATA TO NUMERIC-NONINTEGER-DATA
- MOVE NUMERIC-EDITED-DATA TO NUMERIC-EDITED-DATA

لاحظ أنه لا يسمح بالخليط، عندما يكون من عناصر بيانات حرفية أو حرفية عددية، أو منقحة مع عناصر بيانات عددية. يسمح بأى خليط آخر غير مذكور اعلاه .

**مثـال ٦ - ٣٠ :**

كانت كل عبارات النقل الموضحة حتى الآن من عنصر بيانات فردى لعنصر آخر . اعتبر محاولة نقل المجموعة التالية :

```
01  INPUT-INVOICE-RECORD.
    05  INPUT-VENDOR-ID          PIC X(5).
    05  INPUT-INVOICE-ID         PIC X(4).
    05  INPUT-NUMBER-ITEMS       PIC S9(4).
    05  INPUT-AMOUNT             PIC S9(5)V99.

01  PAYABLE-INVOICE-RECORD.
    05  PAYABLE-VENDOR-ID        PIC X(5).
    05  PAYABLE-INVOICE-ID       PIC X(4).
    05  PAYABLE-NUMBER-ITEMS     PIC S9(4)       COMP-3.
    05  PAYABLE-AMOUNT           PIC S9(5)V99    COMP-3.

MOVE INPUT-INVOICE-RECORD TO PAYABLE-INVOICE-RECORD
```

عند نقل مجموعة عناصر ( على اى مستوى )، أو عند استقبال مجموعة عناصر لقيمة منقولة.. فإنها تعامل كعنصر حرفى عددى كبير .

هنا.. لكل الحقول فى INPUT- INVOICE- RECORD استخدام DISPLAY ، ويبلغ طول مجموعة العناصر 20 بايت (5 + 4 + 4 + 7) ونقلت كما لو كانت معرفة بصورة PIC X(20) . يبلغ طول PAYABLE- INVOICE- RECORD 16 بايت (5 + 4 + 3 + 4). لاحظ الفرق الذى يحدث بسبب COMP-3 ، المستخدمة مع العناصر العددية ) ، وعلى هذا – ولأغراض النقل – فإنه يعامل كما لو كان معرفا بالصورة PIC X(16) . أما تنفيذ عبارة النقل.. فإنه.

( ١ ) لايحول الحقول الجزئية INPUT- NUMBER- ITEMS , INPUT- AMOUNT من DISPLAY إلى COMP-3، قبل نقلها على التوالى إلى PAYABLE- AMOUNT , PAYABLE - NUMBER - ITEMS .

( ٢ ) يحذف الأربع بايت الموجودة فى أقصى يمين INPUT- INVOICE- RECORD [ الفرق بين PIC X , PICC X(16) (20) ] . النتيجة هى نفايا موجودة فى PAYABLE - NUMBER - ITEMS , PAYABLE- AMOUNT التى تتسبب فى نهاية البرنامج نهاية غير طبيعية إذا ما أشير لهذه الحقول .

**مثـال ٦ – ٣١ :**

نستدل من مثال ٦ – ٣٠ على أن نقل مجموعة غير آمن إلا إذا حدث وكان :

( ١ ) لكل الحقول الجزئية فى كل من العنصر الراسل والمستقبل استخدام DISPLAY ، كما أنه يتوقع حدوث أى تأثير نتيجة الاختلافات فى الأطوال، أو فى ترتيب الحقول الجزئية من قبل المبرمج .

**أو ، ايضا**

( ٢ ) لاتعد كل الحقول الجزئية من نوع DISPLAY، ولكن العنصر الراسل والعنصر المستقبل يناظران بعضهما البعض تماما من ناحية مواقع الحقول الجزئية كلها والصورة والاستخدام

اعتبر مجموعتى العناصر التاليتين :

```
01  PENDING-INVOICE-RECORD.
    05  PENDING-INVOICE-ID          PIC X(4).
    05  PENDING-VENDOR-ID           PIC X(5).
    05  PENDING-AMOUNT-DUE          PIC S9(5)V99.
    05  PENDING-NUMBER-ITEMS        PIC S9(4).

01  PAYABLE-INVOICE-RECORD.
    05  PAYABLE-VENDOR-ID           PIC X(5).
    05  PAYABLE-AMOUNT-DUE          PIC S9(5)V99.
    05  PAYABLE-INVOICE-ID          PIC X(4).
    05  PAYABLE-NUMBER-ITEMS        PIC S9(4).
```

حيث إن الشرط الأول سالف الذكر لم يتحقق.. فإن :

```
MOVE PENDING-INVOICE-RECORD TO PAYABLE-INVOICE-RECORD
```

لاتعمل . بسبب الترتيب المختلف للحقول الجزئية فى مجموعتى العناصر.. ينقل PENDING- INVOICE- ID (4 بايت)، وأول بايت من PENDING- INVOICE- ID الى PAYABLE- INVOICE- ID ( 5 بايت ) ... الخ . وتنتج عن هذا بالطبع- أخطاء عندما يشار الى حقول PAYABLE- INVOICE- RECORD .

الطريقة الصحيحة لتحقيق النتائج المرجوة، هى نقل الحقول الجزئية حقلا حقلا : .

```
MOVE  PENDING-INVOICE-ID      TO  PAYABLE-INVOICE-ID
MOVE  PENDING-VENDOR-ID       TO  PAYABLE-VENDOR-ID
MOVE  PENDING-AMOUNT-DUE      TO  PAYABLE-AMOUNT-DUE
MOVE  PENDING-NUMBER-ITEMS    TO  PAYABLE-NUMBER-ITEMS
```

( فى بعض المؤسسات.. يمكن أن يتأثر نقل مجموعات معينة بعبارة نقل على النحو التالى : MOVE CORRESPONDING identifier-1 to identifier- 2 ، إلا أننا نوصى – بشدة – باستخدام نقل العناصر الفردية بدلا من ذلك ) .

## ٦ - ٩ إيقاف تنفيذ البرنامج :

### عبـــارة التوقـــف

### إيقاف التنفيذ نهائيا

تتسبب عبارة التنفيذ STOP RUN فى أن يعيد برنامج الهدف تحكم الكمبيوتر الى نظام التشغيل . ويفصل هذا الإجراء هذا التنفيذ للبرنامج تماما عن الكمبيوتر. ويجب أن تغلق كل الملفات قبل تنفيذ STOP RUN وإلا فإنه يمكن أن تحدث أخطاء جسيمة أثناء تشغيل لاحق للملف . يجب أن توجد فى كل برنامج عبارة STOP RUN واحدة على الأقل ، يتطلب عديد من نمطيات الكتابة وجود عبارة STOP RUN واحدة فقط .

## إيقاف التنفيذ لحظيا

صيغة ايقاف التنفيذ لحظيا هى STOP literal ,وتؤدى مايلى :

( ١ ) عرض قيمة الثابت literal عن طريق النهاية الطرفية التى يستخدمها مشغل الكمبيوتر.

( ٢ ) إيقاف تنفيذ برنامج الهدف لحظيا ( بينما يقوم المشغل بتغيير شرائط أو صيغ ورق الطابع، أو غيرها ) حتى يعطى مشغل الكمبيوتر إشارة لنظام التشغيل عن طريق نهايته الطرفية ، وذلك بالاستمرار فى تنفيذ برنامج الهدف .

مثـــال ٦ - ٣٢ :

```
PROCEDURE DIVISION.
    OPEN  INPUT  PAYABLE-INVOICE-FILE
    STOP "PLEASE MOUNT PREPRINTED CHECK FORMS IN PRINTER"
    OPEN  OUTPUT  PAID-INVOICE-CHECK-FILE
```

بعد قراءة الرسالة ...PLEASE MOUNT على النهاية الطرفية.. يفترض أن يكون مشغل الكمبيوتر قد أعد الطابع لطباعة شيكات ، وعند ذلك يعيد بدء البرنامج عند ...OPEN OUTPUT .

## ٦ - ١٠ تنفيذ متحكم فيه لمقطع :

### عبـــارة التنفيـــذ

تسمح عبارة التنفيذ PERFORM بتنفيذ محتويات مقطع مرة واحدة أو أكثر من أى نقطة فى البرنامج تختلف عن الترتيب التتابعى الذى تنفذ فيه عبارات البرنامج . والصيغة البسيطة لعبارة التنفيذ، هى كما يلى :

<u>PERFORM</u> paragraph-name

حيث إن paragraph- name هو اسم مقطع يعرفه المبرمج فى جزء الإجراءات .. فإن تنفيذ هذه الصيغة يتسبب فى تنفيذ كل العبارات الموجودة فى المقطع المحدد، فى ترتيب طبيعى ( الأولى فالثانية وهكذا ) . يستمر التنفيذ المتتالى الطبيعى للتعليمات بعد ذلك بالعبارات التى تلي عبارة PERFORM مباشرة .

مثـــال ٦ - ٣٣ :

```
PROCEDURE DIVISION.

    OPEN    INPUT     TIME-CARD-FILE
            OUTPUT    TIME-CARD-LISTING

    PERFORM INITIALIZE-COUNTERS-SWITCHES

    WRITE LISTING-LINE
        FROM WS-LISTING-LINE-HEADING
        AFTER ADVANCING PAGE
```

```
    PERFORM INPUT-TIME-RECORD

    MOVE WS-TIME-CARD-ID TO WS-LISTING-ID
    ..............................
INITIALIZE-COUNTERS-SWITCHES.

    MOVE "F" TO TIME-CARD-END-SW
    MOVE ZEROS TO WS-EMPLOYEE-COUNTER
                  WS-AMOUNT-OVERTIME
                  WS-TOTAL-HOURS
    INPUT-TIME-RECORD.

        READ TIME-CARD-FILE
            INTO WS-TIME-CARD-RECORD
            AT END
                MOVE "T" TO TIME-CARD-END-SW
```

بوجود عبارتين PERFORM فإن ترتيب التنفيذ السابق يكون كما يلى :

| | |
|---|---|
| OPEN... ( ١ ) | PERFORM INITIALIZE... ( ٢ ) |
| MOVE " F "... ( ٣ ) | MOVE ZEROS... ( ٤ ) |
| WRITE... ( ٥ ) | PERFORM INPUT-... ( ٦ ) |
| READ ( ٧ ) | MOVE WS_... ( ٨ ) |

تحتاج معظم برامج الأعمال إلى تكرار دورة المدخلات والتشغيل والمخرجات لكل سجل منطقى من سجلات ملف المدخلات . هذا يعنى أن منطق البرنامج program logic يجب أن يحتوى على دورة loop ؛ حيث تنفذ مجموعة من التعليمات مرات ومرات . وعبارة الكوبل الأكثر فائدة فى دورات البرمجة لها التكوين التالى:

<u>PERFORM</u> paragraph-name <u>UNTIL</u> condition

عندما تنفذ مثل هذه العبارة.. ينفذ المقطع المحدد مرات ومرات حتى يتحقق الشرط المحدد . وتناقش الشروط بالكامل فى الفصل السابع من الكتاب ، والشرط حاليا له الشكل التالى data- name EQUAL literal .

**مثــال ٦ – ٣٤ :**

```
PROCEDURE DIVISION.

    OPEN    INPUT     TIME-CARD-FILE
            OUTPUT    TIME-CARD-LISTING

    MOVE "F" TO TIME-CARD-END-SW

    PERFORM PRODUCE-TIME-LISTING-LINE
        UNTIL TIME-CARD-END-SW EQUAL "T"
```

```
CLOSE TIME-CARD-FILE
        TIME-CARD-LISTING

 STOP RUN


PRODUCE-TIME-LISTING-LINE.

 READ TIME-CARD-FILE
     INTO WS-TIME-RECORD
     AT END
         MOVE "T" TO TIME-CARD-END-SW


 IF TIME-CARD-END-SW EQUAL "F"

 MOVE WS-TIME-REC-ID      TO  WS-LISTING-ID
 MOVE WS-TIME-REC-NAME    TO  WS-LISTING-NAME
 MOVE WS-TIME-REC-HOURS   TO  WS-LISTING-HOURS

 WRITE TIME-CARD-LISTING-LINE
     FROM WS-LISTING-LINE
     AFTER ADVANCING 2 LINES
```

دعنا نفترض أنه لايوجد سوى سجلين منطقيين فقط فى الملف TIME- CARD- FILE . والترتيب المنطقى الذى تنفذ به العبارات، هو مايلى :

(١) OPEN...

(٢) MOVE "F" TO TIME-CARD-END-SW

(٣) PERFORM PRODUCE-TIME-LISTING-LINE UNTIL....

( حيث TIME- CARD- END-SW ليس صحيحا (T) ، فينفذ المقطع لأول مرة )

**اول تنفيذ المقطع**

(٤) READ TIME-CARD-FILE... AT END MOVE "T" TO TIME-CARD-END-SW

( إدخال أول سجل منطقى.. يظل المفتاح "F" )

(٥) IF TIME-CARD-END-SW EQUAL "F"

( حيث أن الشرط صحيح.. فتنفذ كل العبارات حتى النقطة ) .

(٦) MOVE WS-TIME-REC-ID...

(٧) MOVE WS-TIME-REC-NAME...

(٨) MOVE WS-TIME-REC-HOURS..

(٩) WRITE TIME-CARD-LISTING-LINE FROM...

(بعد آخر عبارة فى المقطع.. يتم تقويم الشرط. وحيث أن المفتاح ليس "T" يعاد تنفيذ المقطع مرة ثانية)

**التنفيذ الثانى للمقطع**

(١٠) READ TIME-CARD-FILE... AT END MOVE "T" TO TIME-CARD-END-SW

( إدخال ثانى منطقى ، يظل المفتاح "F" )

(١١) IF TIME-CARD-END-SW EQUAL "F"

(حيث إن الشرط صحيح، تنفذ كل العبارات حتى النقطة .)

(١٢) MOVE WS-TIME-REC-ID...

(١٣) MOVE WS-TIME-REC-NAME...

(١٤) MOVE WS-TIME-REC-HOURS...

(١٥) WRITE TIME-CARD-LISTING-LINE FROM...

(بعد آخر عبارة فى المقطع.. يتم تقويم الشرط. وحيث إن المفتاح ليس "T" يعاد تنفيذ المقطع مرة ثانية)

(١٦) READ TIME-CARD-FILE... AT END MOVE "T" TO TIME-CARD-END-SW

التنفيذ الثالث للمقطع

( عند READ الثالثة للملف الموجود فيه سجلان منطقيان فقط.. تحدث نهاية الملف .. ينفذ ...AT END جاعلا المفتاح ، محتويا على "T" ، وتستمر PERFORM حتى نهاية المقطع ).

( ١٧ ) IF TIME-CARD-END-SW EQUAL "F"

( المفتاح الآن "T" ، وتترك كل العبارات حتى النقطة التى تنهى ...IF. هذه النقطة تنهى المقطع كذلك ، جاعلة ...IF هى آخر عبارة تنفذ فى البرنامج ) .

( ١٨ ) CLOSE...

( بعد تنفيذ آخر عبارة فى مقطع PERFORM .. يختبر الكمبيوتر المفتاح . وحيث إنه يحتوى على "T" الآن ، فينتهى تنفيذ عبارة PERFORM وتنفذ العبارة التالية ...CLOSE ).

( ١٩ ) STOP RUN

( بعد تنفيذ STOP RUN.. لاتنفذ أى عبارة أخرى من عبارات البرنامج ) .

لاحظ كيف يتجنب منطق البرنامج – بعناية تشغيل سجل بعد حدوث نهاية الملف ( ارجع الى مثال ٦ – ١٠ ) .

**مثـال ٦ – ٣٥ :**

بعد توضيح عمل الدورات.. نكون الآن فى موقع يسمح لنا بفحص برنامج كوبل كامل.. اعتبر تطبيق الرواتب الذى يملأ فيه العاملون استمارات وقت، طبقا لأرقام تعريفهم وأسمائهم، وإجمالى عدد ساعات عملهم الأسبوعية . يوقع الملاحظ استمارات الوقت هذه، ويرسلها إلى تشغيل البيانات ؛ حيث يتم إدخال المعلومات باستخدام معدات من لوحة المفاتيح إلى القرص مباشرة key- to- disk . والمرغوب فيه وجود برنامج تنقيح edit program ، يطبع نسخة دائمة من سجلات الوقت الأسبوعية لفحصها بصريا وتصحيحها .

قائمة مصدر مثل هذا البرنامج مدونة أدناه . ونأمل أن يكون القارئ قادرا على تعريف التطبيقات لأساسيات برمجة الكوبل التى سبق مناقشتها حتى الآن . وبصفة خاصة :

( ١ ) أجزاء ORGANIZATION و ACCESS فى عبارات SELECT ، المسهب فيها ( حيث ان الحالة التقليدية هى SE QUENTLAL) ، والتى تحدد لتوثيق البرنامج .

( ٢ ) استخدام أجهزة كمبيوتر IBM لجزء BLOCK CONTAINS 0 للفات مدخلات أقراص .

( ٣ ) حذف جزء BLOCK CONTAINS من ملف الطباعة .

( ٤ ) استخدام LABEL RECORDS ARE STANDARD لملف القرص ، واستخدام OMITTED لملف الطابع .

( ٥ ) إجراء كل تشغيل السجل فعلا فى مخزن العمل باستخدام READ... INTO و WRITE... FROM ، ويكون وصف السجل فى قسم الملفات مثل PIC X (80) , PI X(132) فقط ، حيث يحدد التخطيط الفعلى للسجل فى نسخة مخزن العمل .

( ٦ ) تجميع عناصر مخزن العمل تحت فئات على المستوى 01 ، وسرد العناصر بطريقة منظمة .

( ٧ ) استخدام جزء VALUE فى عمل كل من أسطر العناوين وأسطر أرقام التعريف، والاسم، والساعات فى مخزن العمل .

( ٨ ) استخدام FILLER لكل عناصر البيانات التى لايشار اليها فى جزء الإجراءات .

( ٩ ) يستخدم جزء الإجراءات عبارات PERFORM , PERFORM UNTIL.... استخداما واسعا للتحكم فى سريان منطق البرنامج وعمل الدورات الأساسية للمدخلات والتشغيل والمخرجات .

(١٠) أقناع نفسك بأن هذا البرنامج لايقوم بتشغيل سجل به نفايا بعد الوصول إلى نهاية الملف ( وأنه يقوم بتشغيل كل السجلات المنطقية فى ملف القرص المحتوى على معلومات استمارات الوقت ) .

(١١) ملاحظة الترحيل فى جزء الإجراءات ، والترحيل وضبط الأعمدة فى جزء البيانات .

```
       IDENTIFICATION DIVISION.

       PROGRAM-ID. HRVERIFY.

       AUTHOR.   LARRY NEWCOMER.
       INSTALLATION.   PENN STATE UNIVERSITY, YORK CAMPUS.

       DATE-WRITTEN.  MAY 1983.
       DATE-COMPILED. MAY  9,1983.

       SECURITY.  NONE.

      *    HRVERIFY PRODUCES A PRINTED LISTING OF THE WEEKLY PAYROLL
      *    TIME FILE (PREPARED BY KEY-TO-DISK OPERATORS USING WEEKLY
      *    TIME SHEETS).  THIS LISTING IS USED TO VISUALLY VERIFY
      *    THE CORRECTNESS OF EMPLOYEE ID NUMBERS, EMPLOYEE NAMES,
      *    AND EMPLOYEE WEEKLY HOURS WORKED AS TYPED BY DATA ENTRY.
      *    THE ONLY OUTPUT IS THIS VISUAL EDIT REPORT.


       ENVIRONMENT DIVISION.

       CONFIGURATION SECTION.

       SOURCE-COMPUTER.   IBM-3081.
       OBJECT-COMPUTER.   IBM-3081.

       INPUT-OUTPUT SECTION.

       FILE-CONTROL.

           SELECT TIME-SHEET-INPUT-FILE
               ASSIGN TO TIMECARD
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
               .

           SELECT VERIFY-TIME-REPORT
               ASSIGN TO TIMELIST
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
               .


       DATA DIVISION.

       FILE SECTION.

       FD  TIME-SHEET-INPUT-FILE
           BLOCK CONTAINS 0 RECORDS
           RECORD CONTAINS 80 CHARACTERS
           LABEL RECORDS ARE STANDARD
           .

       01  TIME-SHEET-INPUT-RECORD      PIC X(80).

       FD  VERIFY-TIME-REPORT
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED
           LINAGE IS 50
               WITH FOOTING AT 45
               LINES AT TOP 8
               LINES AT BOTTOM 8
           .

       01  VERIFY-TIME-LINE             PIC X(132).
```

```
       WORKING-STORAGE SECTION.

        01  PROGRAM-FLAGS-AND-SWITCHES.
            05  WS-TIME-SHEET-END-SW     PIC X.

        01  WORKING-INPUT-RECORD-AREAS.

            05  WS-TIME-SHEET-REC.
                10  WS-TIME-SHEET-ID     PIC X(4).
                10  WS-TIME-SHEET-NAME   PIC X(20).
                10  WS-TIME-SHEET-HOURS  PIC S9(2)V9.
                10  FILLER               PIC X(53).

        01  WORKING-OUTPUT-RECORD-AREAS.

            05  WS-VERIFY-HEADING.
                10  FILLER               PIC X(4)         VALUE "-ID-".
                10  FILLER               PIC X(4)         VALUE SPACES.
                10  FILLER               PIC X(20)        VALUE "NAME".
                10  FILLER               PIC X(4)         VALUE SPACES.
                10  FILLER               PIC X(5)         VALUE "HOURS".
                10  FILLER               PIC X(95)        VALUE SPACES.

            05  WS-VERIFY-LINE.
                10  WS-VERIFY-ID         PIC X(4).
                10  FILLER               PIC X(4)         VALUE SPACES.
                10  WS-VERIFY-NAME       PIC X(20).
                10  FILLER               PIC X(4)         VALUE SPACES.
                10  WS-VERIFY-HOURS      PIC Z9.9-.
                10  FILLER               PIC X(95)        VALUE SPACES.


       PROCEDURE DIVISION.

           PERFORM INITIALIZE-FILES-AND-SWITCHES

           PERFORM PRODUCE-VERIFY-LINE
               UNTIL WS-TIME-SHEET-END-SW  EQUAL "T"

           CLOSE   TIME-SHEET-INPUT-FILE
                   VERIFY-TIME-REPORT


           STOP RUN
           .

       INITIALIZE-FILES-AND-SWITCHES.

           MOVE "F" TO WS-TIME-SHEET-END-SW
           OPEN    INPUT   TIME-SHEET-INPUT-FILE
                   OUTPUT  VERIFY-TIME-REPORT
           PERFORM READ-TIME-SHEET-FILE
           WRITE VERIFY-TIME-LINE
               FROM WS-VERIFY-HEADING
               AFTER ADVANCING PAGE
           .

       READ-TIME-SHEET-FILE.

           READ TIME-SHEET-INPUT-FILE
               INTO WS-TIME-SHEET-REC
               AT END
                   MOVE "T" TO WS-TIME-SHEET-END-SW
           .
```

```
          PRODUCE-VERIFY-LINE.

              MOVE WS-TIME-SHEET-ID        TO WS-VERIFY-ID
              MOVE WS-TIME-SHEET-NAME      TO WS-VERIFY-NAME
              MOVE WS-TIME-SHEET-HOURS     TO WS-VERIFY-HOURS
              WRITE VERIFY-TIME-LINE
                  FROM WS-VERIFY-LINE
                  AFTER ADVANCING 3 LINES
                  AT END-OF-PAGE
                      WRITE VERIFY-TIME-LINE
                          FROM WS-VERIFY-HEADING
                          AFTER ADVANCING PAGE
              .
              PERFORM READ-TIME-SHEET-FILE
              .
```

ويمكن أن تشبه المخرجات من هذا البرنامج مايلي :

```
0027    FOLKERS DEIRDRE        45.6
```

## ٦ - ١١ تشغيل : عبارة الجمع

تتسبب عبارة الجمع ADD فى جمع محتويات عنصرين بيانات ( أو أكثر ) من النوع العددى، ووضع حاصل الجمع فى واحد ( أو أكثر ) من عناصر البيانات هذه . وهناك صيغتان رئيسيتان لهذه العبارة ، وهما : ...ADD , ...ADD... GIVING ... TO ( ولاتوصى بصيغة ثالثة ADD CORRESPONDING مع حذفها من هذا الكتاب ) .

## اضـــف ... إلـــى

هذا البديل ... ADD... TO ، له التكوين المبين فى شكل (٦ - ٤) .

ADD {identifier-1 / literal-1} {identifier-2 / literal-2} . . .

TO identification-m [ROUNDED] [identification-n [ROUNDED]]

[ON SIZE ERROR imperative-statement]

شكل (٦ - ٤)

مثـــال (٦ - ٣٦) :

- ADD WEEKLY-HOURS TO TOTAL-HOURS

تضاف المحتويات الحالية للعنصر WEEKLY- HOURS الى المحتويات الحالية للعنصر TOTAL- HOURS ويوضع حاصل الجمع فى TOTAL- HOURS ، ولاتتغير محتويات WEEKLY- HOURS . يجب أن يكون كل من العنصرين عدديا (وليس منقحا ) . وينتج المترجم – بصورة تلقائية – لغة آلة إضافية لمعالجة ضبط العلامات العشرية، وتحويل الصورة من استخدام USAGE معين لاستخدام آخر .

- ADD ITEM-A ITEM-B TO ITEM-C

تضاف محتويات العناصر TIEM- A , ITEM- B , ITEM- C مع بعضها ، ويوضع حاصل الجمع فى العنصر ITEM- C ، وتظل كل من محتويات TIEM- A , ITEM- B كما هى دون تغيير. يجب أن تكون كل عناصر البيانات عددية (وليست منقحة).

- ADD ITEM-A ITEM-B ITEM-C TO ITEM-D ITEM-E

يوضع مجموع العناصر TIEM- A , ITEM- B , TIEM- C , ITEM- D فى العنصر ITEM- D ، ومجموع العناصر TIEM- A , ITEM- B , TIEM- C و TIEM- E يوضع فى TIEM- E . تظل قيم TIEM- C , ITEM-B , TIEM- A كما هى دون تغيير .

- ADD 1 TO NUMBER-OVERTIME-EMPLOYEES

يضاف 1 الى المحتويات الحالية للعنصر NUMBER- OVERTIME- EMPLOYEES ( والذى يتزايد على ذلك بمقدار ( 1 ) . استخدام الثوابت العددية معتاد فى الحسابات .

## اضـف ... معطـيا

لهذا البديل ADD... GIVING... التكوين المبين فى شكل (٦ - ٥) .

```
ADD {identifier-1} {identifier-2} [identifier-3] ...
    {literal-1   } {literal-2   } [literal-3   ]
    GIVING identifier-m [ROUNDED] [identifier-n [ROUNDED]] ...
    [ON SIZE ERROR imperative-statement]
```

شكل (٦ - ٥)

مثـال ٦ - ٣٧ :

- ADD REGULAR-HOURS OVERTIME-HOURS GIVING TOTAL-HOURS

يتطلب بديل GIVING عنصرى بيانات على الأقل ع ـ يسار كلمة GIVING وعنصر بيانات واحد على الأقل على يمينها. تضاف المحتويات الفعلية لكل من REGULAR- HOURS و OVERTIME- HOURES ، ويوضع حاصل الجمع فى TO- TAL - HOURS . وتظل محتويات REGULAR- HOURS و OVERTINE- HOURS كما هى دون تغيير. ولاتدخل المحتويات الاصلية للعنصر TOTAL - HOURS في عملية الجمع . يجب أن تكون عناصر البيانات الموجودة على يسار GIV- ING ، والتى تجمع مع بعضها البعض، عددية وغير منقحة. تستقبل عناصر البيانات الموجودة على يمين GIVING نتيجة الجمع ولاتجمع هى بنفسها ، وعلى هذا.. فبالرغم من أنها يجب أن تكون عددية إلا أنها يمكن أن تكون منقحة .

```
ADD  PAYABLE-INVOICE-AMOUNT
     PAYABLE-INVOICE-FREIGHT
     PAYABLE-INVOICE-SPECIAL  GIVING  OUTPUT-CHECK-AMOUNT
```

تسهل طريقة الكتابة من القراءة ومن إجراء التعديلات . يجب أن تكون كل حقول PAYABLE- INVOICE عددية وغير منقحة . تجمع الحقول ويوضع حاصل جمعها بدلا من محتويات OUTPUT- CHECK- AMOUNT (الذى لايدخل فى الجمع)، ويكون الحقل OUTPUT- CHECK- AMOUNT منقحا للطباعة .

```
ADD  STATE-TAX
     FEDERAL-TAX  GIVING  PAYROLL-REPORT-TOTAL-TAX
                          EMPLOYEE-MASTER-TOTAL-TAX
```

يجب أن تكون كل الحقول عددية ، إلا PAYROLL- REPOT- TOTAL- TAX يكون منقحا ( يسرى هذا عند استقبال نتائج ) . يجمع STATE- TAX و FEDERAL- TAX ويوضع حاصل جمعهما فى PAYROLL- REPORT- TOTAL- TAX وفى EMPLOYEE- MASTER- TOTAL- TAX

مثـال ٦ - ٣٨ :

- ADD STATE-TAX TO FEDERAL-TAX GIVING TOTAL-TAX

خطأ تكوينى : لايمكن استخدام TO ، و GIVING معا .

- ADD FREIGHT-CHARGES TO INVOICE-AMOUNT AND SHIPPING-TOTAL

AND غير صحيحة فى كل من صيغتى ADD

- ADD DAILY-HOURS GIVING WEEKLY-HOURS MONTHLY-HOURS

تتطلب صيغة GIVING قيمتين على الأقل، على يسار كلمة GIVING . وربما كان المقصود مايلى :

```
ADD DAILY-HOURS TO   WEEKLY-HOURS   MONTHLY-HOURS
```

يوضح استخدام جزء ROUNDED وجزء ON SIZE ERROR مع كل من الصيغتين للجمع ADD فى القسمين 12 و 13 – الفصل السادس .

## ٦ – ١٢ تشغيل : حذف جزء من الكسر العشرى وجزء التقريب

كل العبارات الحسابية (COMPUTE , DIVIDE , MULTIPLY , SUBTRACT , ADD) تنتج تلقائيا تعليمات بلغة الآلة.

(١) لضبط العلامات العشرية للقيم التى تجمع أو تطرح

(٢) التحويل من أحد الاستخدامات USAGE إلى استخدام آخر والعكس .

من الممكن أن تكون نتيجة الحسابات لها مواقع للكسر العشرى، أكثر من المواقع المحددة له فى الحقل المستقبل . والإجراء التقليدى ، فى هذه الحالة هو حذف العدد اللازم من الأرقام من ناحية اليمين .

لتعديل عملية الحذف ، يمكن وضع كلمة التقريب ROUNDED بعد اسم أى حقل يستقبل النتيجة للعملية الحسابية ( راجع الشكلين رقم ٦ – ٤ ، ٦ – ٥ ) . مع تحديد ROUNDED :

(١) إذا كان الرقم الموجود على أقصى اليسار من الجزء المحذوف ٥ أو أكثر.. يضاف 1 إلى الرقم المجاور له، والموجود فى اليمين ( أى يحدث تقريب ) للأرقام المتبقية .

(٢) أذا كان الرقم الموجود على أقصى اليسار من الجزء المحذوف ٤ أو أقل ، فتحذف النتيجة ببساطة ، دون تغيير للرقم الموجود فى اقصى يمين الأرقام المتبقية .

**مثـال ٦ – ٣٩ :**

```
05  ANSWER     PIC S99V9
. . . . . . . . . . . . . . . . . . . . . . . . . .
ADD  X  Y  Z  GIVING ANSWER
```

إذا كان مجموع X , Y , Z هو 54.34999 فذلك يلغى آخر اربع مواقع للكسر العشرى فى النتيجة، ويحتوى العنصر ANSWER على 54.3 .

```
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
ADD X   Y   Z   GIVING ANSWER ROUNDED
```

بمواصفة ROUNDED .. فإننا نحصل على 54.3 فى RESVLT ، أخر رقم يحذف هو 4 ، ولاتقرب النتيجة إلى 54.4.

**مثـال ٦ – ٤٠ :**

```
ADD HOURS-WORKED TO WEEKLY-TOTAL
                    MONTHLY-TOTAL  ROUNDED
                    YEARLY-TOTAL   ROUNDED
```

من المسموح به تحديد ROUNDED لبعض الحقول المستقبلة فقط فى الحسابات .

## ٦ – ١٣ تشغيل : السريان الزائد وجزء عند حدوث خطأ فى الحجم

إذا كان جزء الرقم الصحيح integer part للحقل المستقبل أقل من القيمة الخاصة بالنتيجة فسوف يحدث سريان زائد over flow ، أو شرط خطأ فى الحجم SIZE ERROR . ويسمح جزء ON SIZE ERROR للمبرمج بأن يحدد مجموعة من العبارات التى لاتنفذ إلا إذا تحقق شرط SIZE ERROR .

**مثـال ٦ – ٤١ :**

```
ADD WEEKLY-HOURS TO YEARLY-TOTAL-HOURS
    ON SIZE ERROR
        MOVE EMPLOYEE-ID    TO ERROR-LINE-ID
        MOVE WEEKLY-HOURS TO ERROR-LINE-HOURS
        MOVE "YEARLY HOURS OVERFLOW--HOURS NOT INCLUDED IN TOTAL"
            TO ERROR-LINE-MESSAGE
        WRITE REPORT-LINE
            FROM ERROR-LINE
            AFTER ADVANCING 3 LINES
.
PERFORM COMPUTE-GROSS-PAY
...................................
```

إذا كانت نتيجة جمع YEARLY- TOTAL- HOURS , WEEKLY- HOURS متفقة مع YEARLY- TOTAL- HOURS فتهمل عبارات ON SIZE ERROR بالنسبة لتنفيذ عبارة ADD . لاحظ ان جزء ON SIZE ERROR (مثل AT END... , END- OF- PAGE...) ينتهى بنقطة . يتسبب حذف النقطة المرتبة التى تسبق عبارة PERFORM فى جعل عبارة PERFORM جزءا من جزء ON SIZE ERROR ، ونتيجة ذلك هى تنفيذ PERFORM عند تحقق شرط ON SIZE ERROE فقط ؛ بدلا من تنفيذها فى كل مرة مطلوبة .

**مثـال ٦ – ٤٢ :**

```
    05  A   PIC S99V99   VALUE +50.34.
    05  B   PIC S99V99   VALUE +50.23.
    05  C   PIC S99V9    VALUE +12.3.

●   ADD  A  B  GIVING C
```

حيث إن مجموع A , B هو 100.57+ .. فإن الجزء الصحيح للنتيجة يكون أكبر من نظيره فى الحقل المستقبل C . وبدون جزء ON SIZE ERROE ، يحذف الكمبيوتر – ببساطة – جزء صحيحاً من النتيجة ( من ناحية اليسار ) ويستمر فى العمل . وعلى هذا يحتوى C على 00.5+ ، وهى نتيجة غير صحيحة بالطبع .

```
•    ADD  A  B  GIVING C
         ON SIZE ERROR
            DISPLAY "SIZE ERROR FOR C--C IS UNCHANGED"
```

بتحديد جزء ...ON SIZE ERROE .. لاينقل المجموع المحذوف منه إلى C وتستمر C فى أحتوائها على 12.3+ .أكثر من ذلك ، تطبع عبارة DISPLAY رسالة خطأ ( على ملف نظام خاص SYSOUT بالنسبة لكوبل IBM OS/VS ) .

## خطوط إرشادية خاصة باستخدام عند حدوث خطأ فى الحجم

( ١ ) إذا كان ممكنا ، حدد حجم PIC لعناصر البيانات العددية، بحيث يكون كبيرا بدرجة كافية، بحيث تسمح بالنسبة إلى الحسابات التى يجريها البرنامج ، أن يكون السريان الزائد مستحيلا رياضيا، ويسمح ذلك بحذف جزء ON SIZE ERROE دون خطورة .

( ٢ ) إذا كانت هناك إمكانية بسيطة لحدوث خطأ فى الحجم ، استخدم جزء ON SIZE ERROE .

( ٣ ) عند استخدام ...ON SIZE ERROE يجب على الأجزاء التى تلى حدوث الشرط أن :

(أ) تطبع رسالة بها معلومات كافية، تسمح للشخص المستخدم للكمبيوتر أن يعرف ماحدث .

(ب) ترتب العمل بحيث يترك تشغيل السجلات المنطقية الذى تسبب فى حدوث خطأ الحجم جزئيا أو كليا .

(ج) تسمح باستمرار التشغيل المعتاد لبقية السجلات المنطقية ( لايتوقف تنفيذ البرنامج كله ) .

## ٦ – ١٤ تشغيل : عبارة الطرح

يعطى شكل ٦ – ٦ صيغتين رئيسيتين لعبارة الطرح SUBTRACT . ونظرا لأن ...SUBTRACT... FROM تناظر فى كل الوجوه ...ADD... TO فنحتاج الى توضيح معالجة ...SUBTRACT... FROM... GIVING فقط .

```
SUBTRACT {identifier-1} [identifier-2] ...
         {literal-1   } [literal-2   ]
    FROM identifier-m [ROUNDED] [identifier-n [ROUNDED]] ...
    [ON SIZE ERROR imperative-statement]
```

(*a*)

```
SUBTRACT {identifier-1} [identifier-2] ...
         {literal-1   } [literal-2   ]
    FROM {identifier-m}
         {literal-m   }
    GIVING identifier-n [ROUNDED] [identifier-o [ROUNDED]] ...
    [ON SIZE ERROR imperative-statement]
```

(*b*)

شكل (٦ – ٦)

**مثـال ٦ - ٤٣ :**

```
•   SUBTRACT DISCOUNT-AMOUNT FROM CURRENT-PRICE GIVING NEW-PRICE
```

تطرح المحتويات الحالية للعنصر DISCOUNT- AMOUNT من المحتويات الحالية للعنصر CURRENT- PRICE ، يوضع باقى الطرح فى NEW- PRICE . يمكن أن يكون الحقل NEW- PRICE حقلا منقحا؛ حيث إنه لايدخل بنفسه فى الحسابات . ولايتأثر أى من DISCOUNT- AMOUNT او CURRENT- PRICE بعملية الطرح .

```
•   SUBTRACT 10 FROM A-GRADE-CUTOFF GIVING B-GRADE-CUTOFF
```

عادة ماتستخدم الثوابت العددية فى العبارات الحسابية .

```
•   SUBTRACT  STATE-TAX
              FEDERAL-TAX
              FICA-AMOUNT  FROM GROSS-PAY
              GIVING NET-PAY  ROUNDED
```

NET- PAY هو عنصر البيانات الوحيد الذى تتغيير محتوياته . ويطرح مجموع FICA- AMOUNT , FEDER- AL- TAX , STATE- TAX من GROSS- PAY ، ويوضع باقى الطرح فى NET- PAY بعد تقريبه .

```
•   SUBTRACT  A  B  C
      FROM D
      GIVING  E  F  ROUNDED  G
      ON SIZE ERROR
         PERFORM OVERFLOW-ROUTINE.
```

يطرح مجموع A , B , C من D ، وتوضع نسخة من النتيجة فى E ، وتقرب نسخة من النتيجة وتوضع فى F ، وتوضع نسخة فى G . إذا تحقق شرط حدوث خطأ فى الحجم أثناء أى من هذه العمليات..فإن المقطع المسمى OVERFLOW- ROU- TINE ينفذ مرة واحدة ، قبل أن يستمر الكمبيوتر فى تنفيذ العبارة التالية .

```
•   01  SAMPLE-DATA.
        05  A     PIC S9(3)V9.
        05  B     PIC S9V99      COMP-3.
        05  C     PIC ZZZ.99.

    SUBTRACT B FROM C GIVING A
```

الطرح غير صحيح ، لأنه يحاول شمول عنصر منقح (C) فى الحسابات .

- SUBTRACT A B FROM C D GIVING E F

خطأ تكوينى : يمكن لعنصر بيانات واحد فقط أن يتبع كلمة FROM .

## ٦ - ١٥ تشغيل : عبارة الضرب

تستخدم عبارة الضرب MULTIPLY فى ضرب عنصرين عدديين ، وللعبارة صيغتان .

### اضرب فى

تكوين عبارة أضرب فى ...MULTIPY... BY مبين فى شكل (٦ - ٧) .

```
MULTIPLY {identifier-1} BY identifier-2 [ROUNDED] [identifier-3 [ROUNDED]] . . .
         {literal-1   }
    [ON SIZE ERROR imperative-statement]
```

شكل (٦ - ٧)

مثال ٦ - ٤٤ :

- MULTIPLY QUANTITY-PURCHASED BY UNIT-COST

تضرب المحتويات الحالية للعنصر QUANTITY- PURCHASED فى المحتويات الحالية للعنصر UNIT- COST ويخزن فيه حاصل الضرب، ولا يتغير QUANTITY- PURCHASED .

```
MULTIPLY QUANTITY-PURCHASED
    BY UNIT-COST ROUNDED
    ON SIZE ERROR
        PERFORM EXTENDED-COST-TOO-LARGE
```

يقرب حاصل الضرب قبل وضعه فى UNIT- COST ، إذا حدث خطأ فى الحجم .. ينفذ المقطع -EXTENDED COST- TOO- LARGE مرة واحدة ، قبل أن يستمر الكمبيوتر فى تنفيذ العبارة التى تلى عبارة MULTIPLY .

- MULTIPLY .90 BY WHOLESALE-PRICE
  RETAIL-PRICE ROUNDED

تضرب محتويات WHOLESALE- PRICE فى 90.. وتوضع النتيجة فى WHOLESALE- PRICE ( مع حذف مواقع الكسر العشرى الزائدة ) ، ثم تضرب محتويات RETAIL- PRICE فى 90، ويوضع حاصل الضرب بعد تقريبه – فى -RE TAIL- PRICE .

- MULTIPLY CURRENT-PRICE BY .90

هذا خطأ شائع جدا . ولسوء الحظ.. فإن هذا يوجه الكمبيوتر لضرب محتويات CURRENT- PRICE فى 90.. يعتبر وضع النتيجة فى الثابت 90 مستحيلاً ، وتصحيح ذلك هو مايلى :

MULTIPLY .90 BY CURRENT-PRICE

بافتراض أن المبرمج لايهمه شيئًا فى فقدان المحتويات الاصلية للعنصر CURRENT- PRICE .

## اضـرب .. فـى .. معطـيا ..

تكوين اضرب .. فى .. معطيا .. MULTIPLY.. BY.. GIVING مبين فى شكل (٦ – ٨) .

```
MULTIPLY {identifier-1} BY {identifier-2}
         {literal-1   }    {literal-2   }
    GIVING identifier-3 [ROUNDED] [identifier-4 [ROUNDED]] . . .
    [ON SIZE ERROR imperative-statement]
```

شكل ( ٦ – ٨)

مثـال ٦ – ٤٥ :

- MULTIPLY UNIT-PRICE BY QUANTITY-PURCHASED
  GIVING EXTENDED-PRICE

تضرب المحتويات الحالية لكل من : QUANTITY- PURCHASED , CURRENT- PRICE ويوضع حاصل الضرب فى EXTENDED- PRICE ، ولاتتغير بقية عناصر البيانات الاخرى . يجب ان تكون كل عناصر البيانات عددية ، ويمكن أن يكون EXTENDED- PRICE منقحا؛ حيث إنه يستقبل نتيجة الحسابات فقط ولايدخل فيها .

```
•    MULTIPLY A BY B
        GIVING C
                D  ROUNDED
                E
        ON SIZE ERROR
            PERFORM FIX-UP-ROUTINE
```

تضرب محتويات A فى محتويات B ، ويوضع حاصل الضرب في C ( مع حذف مواقع الكسر العشرى الأكثر )، وفى D (مع التقريب) وفى E (مع الحذف) . إذا تحقق شرط حدوث خطأ فى الحجم أثناء اى من هذه العلمليات ، فينفذ FIX- UP- RONTINE قبل إنتقال الكمبيوتر إلى العبارة التالية .

```
•    MULTIPLY  HOURS-WORKED  PAY-RATE  BY OVERTIME-FACTOR
        GIVING OVERTIME-PAY
```

خطأ تكوينى : لا يضرب HOURS- WORKED , PAY- RATE فى OVERTIME- FACTOR معا ولايحدث ذلك بالعبارة التالية كذلك .

```
MULTIPLY  HOURS-WORKED  PAY-RATE  BY OVERTIME-FACTOR
```

على عكس ...ADD , ...SUBTRACT. .. يمكن لأمر ...MULTIY أن يضرب قيمتين أثنتين فقط ، احد حلول هذه المشكلة يمكن أن يكون على النحو التالى :

```
MULTIPLY HOURS-WORKED BY PAY-RATE GIVING TEMP
MULTIPLY TEMP              BY OVERTIME-FACTOR
    GIVING OVERTIME-PAY
```

حيث TEMP هى منطقة مخزن عمل يعرفها المبرمج ، وللحصول على حل أفضل.. انظر القسم السابع عشر من هذا الفصل.

## ٦ - ١٦ تشغيل : عبارة القسمة

تستخدم عبارة DIVIDE فى قسمة محتويات أحد عناصر البيانات العددية بمحتويات عنصر بيانات عددى آخر . ولها ثلاثة أشكال: أحدها يعطى نتيجتين: خارج القسمة quotient والباقى remainder . ولاتسمح أى صيغة من صيغ عبارة القسمة بالقسمة على صفر . تتسبب محاولة القسمة على صفر فى إنهاء البرنامج نهاية غير طبيعية، الا إذا ما تحدد جزء ON SIZE ERROR ، حيث يحدث فى مثل هذه الحالة خطأ فى الحجم .

## اقســم .. ب ..

يوضح تكوين اقسم .. بـ ...DIVIDE... INTO في شكل (٦ - ٩).

```
DIVIDE {identifier-1}
       {literal-1   }
    INTO identifier-2 [ROUNDED] [identifier-3 [ROUNDED]] . . .
   [ON SIZE ERROR imperative-statement]
```

شكل (٦ - ٩)

مثـال ٦ - ٤٦ :

- DIVIDE 100 INTO PERCENTAGE-SCRAPPED

يقسم الثابت العددى بالعنصر PERCENTAGE- SCRAPPED ، الذى تستبدل محتوياته بخارج القسمة . ويجب أن يكون العنصر PERCENTAGE- SCRAPPED عدديا وغير منقحا .

```
DIVIDE A INTO B
              C  ROUNDED
              D
    ON SIZE ERROR
       PERFORM ERROR-MESSAGE-ROUTINE
```

يقسم A بالعنصر B ، وتوضع النتيجة ( بعد الحذف من ناحية اليمين ) فى B ، ويقسم A بالعنصر C ، وتوضـع النتيجة ( بعد التقريب ) فى C ، ويقسم A بالعنصر D ، وتوضع النتيجة ( بعد الحذف ) فى D . إذا حدث خطأ فى الحجم فى أى عملية من العمليات السابقة.. فسوف تنفذ عبارات PERFORM وإلا فإنها تهمل .

- DIVIDE A B C INTO D

خطأ تكوينى : يسمح بمعرف واحد فقط على أى جانب من جانبى INTO. ( ليس لعبارة ... MULTIPLY وعبارة -DI VIDE... نفس تكوين عبارة -ADD وعبارة ...SUBTRACT ) .

## اقسم بـ ... أو على ... معطيا

تكوين عبارة اقسم بـ أو على معطيا DIVIDE... INTO/BY... GIVING موضح فى شكل (٦ - ١٠)

```
DIVIDE {identifier-1} {INTO} {identifier-2}
       {literal-1   } {BY  } {literal-2   }
    GIVING identifier-3 [ROUNDED] [identifier-4 [ROUNDED]] . . .
   [ON SIZE ERROR imperative-statement]
```

شكل (٦ - ١٠)

مثـال ٦ – ٤٧ :

- DIVIDE 100 INTO PERCENTAGE-SCRAPPED GIVING FRACTION-SCRAPPED

تقسم قيمة الثابت 100 بالمحتوى الحالى للعنصر PERCENTAGE- SCRAPPED ، وتخزن النتيجة فى العنصر FRACTION- SCRAPPED . يجب أن تكون كل العناصر عددية ، يمكن أن يكون العنصر FRACTION- SCRAPPED منقحا ، حيث أنه لايستخدم فى الحسابات لكنه يستقبل النتيجة فقط .

- DIVIDE PERCENTAGE-SCRAPPED BY 100 GIVING FRACTION-SCRAPPED

متوافقة تماما فى تأثيرها مع العبارة السابقة .

- DIVIDE NUMBER-EMPLOYEES INTO TOTAL-SALARIES

```
DIVIDE NUMBER-EMPLOYEES INTO TOTAL-SALARIES
    GIVING AVERAGE-SALARY
    ON SIZE ERROR
       PERFORM PRINT-ERROR-MESSAGE
```

تقسم المحتويات الحالية للعنصر TOTAL- SALARIES بالمحتويات الحالية للعنصر NUMBER- EMPLOYEES وتوضع النتيجة فى AVERAGE- SALARY (الذى يمكن ان يكون عنصرا منقحا) . لاتتغيير فيمه TOTAL- SALARIES و NUMBER- EMPLOYEE . إذا حدث خطأ فى الحجم .. ينفذ مقطع PRINT- ERROR- MESSAGE .

- DIVIDE B BY A GIVING C

```
DIVIDE B BY A GIVING C
                     D  ROUNDED
                     E
    ON SIZE ERROR
       PERFORM ERROR-ROUTINE
```

تقسم المحتويات الحالية للعنصر B على المحتويات الحالية للعنصر A ، وتوضع النتيجة فى C ( مع حذف جزء من الكسر إذا كانت هناك حاجة لذلك ) ، وفى D ( مع التقريب )، وفى E ( مع الحذف ) . إذا حدث خطأ فى الحجم أثناء أى من هذه العمليات.. فإن المقطع المسمى ERROR- ROUTINE ينفذ. لا تتغيير قيم B , A ، ويمكن أن تكون العناصر E , D , C منقحة إذا كانت هناك رغبة فى ذلك .

- DIVIDE A INTO B ROUNDED GIVING C

غير صحيح : يمكن استخدام ROUNDED مع حقول تستقبل خارج القسمة فقط .

## اقسم بـ .. أو على معطيا... والباقى

صيغة عبارة القسمة هذه ...DIVIDE... INTO/BY... GIVING... REMAINDER موضحة فى شكل (٦ – ١١) .

```
DIVIDE {identifier-1} {INTO} {identifier-2}
       {literal-1   } {BY  } {literal-2   }
    GIVING identifier-3 [ROUNDED]
    REMAINDER identifier-4
    [ON SIZE ERROR imperative-statement]
```

شكل (٦ - ١١)

**مثـال ٦ - ٤٨ :**

- DIVIDE A INTO B GIVING C REMAINDER D

يجب أن تكون كل عناصر البيانات عددية ، كما يمكن أن يكون D , C منقحين . تقسم المحتويات الحالية للعنصر A بمحتويات B ، ويخزن خارج القسمة تخزن في C؛ أما الباقي فيخزن في D . ويعني خارج القسمة نتيجة القسمة بأي عدد خانات محدد للكسر العشري في PIC ، الخاصة بالعنصر C، أما الباقي.. فيعرف على النحو التالي :

الباقي = المقسوم − ( خارج القسمة × القاسم ( المقسوم عليه ) ).

بالنسبة للمثال ، اذا كان A محتويا على 3.3 ، و B محتويا على 10.2 ويمكن للعنصر C أن يحتوي على خانتين للكسر العشري.. فإن C خارج القسمة هي 3.09 والباقي هو 10.2 - (3.3)(3.09) =003.

إذا حدد جزء ROUNDED مع عنصر بيانات GIVING ( الذي يستقبل خارج القسمة ) ، فإن الباقي يحسب قبل حدوث التقريب . وإذا استخدم جزء ON SIZE ERROR ، وحدث خطأ في الحجم في خارج القسمة ( عنصر GIVING ) .. فلن يتغير أي من عنصر بيانات REMAINDER, GIVING ، وينفذ الجزء الخاص بحدوث الخطأ .أما إذا حدث خطأ في الحجم بالنسبة لعنصر REMAINDER فقط ، فإن عنصر GIVING يستبدل بخارج القسمة مع عدم تغير عنصر REMAINDER. يستطيع البرنامج أن يقرر ما إذا كان خارج القسمة أو الباقي هو الذي تسبب في خطأ الحجم، وذلك بالتأكد مما إذا كان حقل خارج القسمة قد تغير (حدث خطأ الحجم قد من خارج القسمة) . تذكر أنه إذا لم يحدد جزء ON SIZE ERROR.. فإن الكمبيوتر سيستمر في تنفيذ العبارة التالية ، حتى عندما تخزن نتائج خاطئة في عنصر GIVING أو عنصر REMAINDER.

- DIVIDE B BY A GIVING C REMAINDER D

هذه العبارة تكافيء تماماً العبارة السابقة .

- DIVIDE 24 INTO TOTAL-HOURS GIVING NUMBER-OF-DAYS
  REMAINDER HOURS-LEFT-OVER

خارج القسمة هو عدد الأيام الكاملة الممثلة بواسطة TOTAL- HOURS ، ويمثل REMAINDER أى جزء متبقيًا من الأيام بالساعات .

- DIVIDE BALANCE-DUE BY AMOUNT-EACH-PAYMENT
  GIVING NUMBER-OF-FULL-PAYMENTS
  REMAINDER AMOUNT-OF-LAST-PAYMENT

فى تطبيقات التسليف التجارية عادة ماتختلف آخر دفعة مدفوعة عن بقية الدفعات . تحسب عبارة DIVIDE عدد الدفعات المعتادة وقيمة الدفعة الأخيرة ( الأقل ) .

- DIVIDE A INTO B GIVING C ROUNDED REMAINDER D ROUNDED

خطأ تكوينى : يمكن استخدام ROUNDED مع عنصر GIVING فقط ، وليس مع عنصر REMAINDER.

- DIVIDE PAYMENT-AMOUNT INTO BALANCE-DUE
  REMAINDER LAST-PAYMENT

هذا خطأ شائع ، يجب استخدام جزء GIVING عندما يتحدد جزء REMAINDER . إذا لم يكن خارج القسمة مهما فعلا ، فيمكن استخدام الوسيلة التالية :

```
DIVIDE PAYMENT-AMOUNT INTO BALANCE-DUE
    GIVING DUMMY-FIELD
    REMAINDER LAST-PAYMENT
```

حيث DUMMY- FIELD معروف فى مخزن العمل ( لكنه لايستخدم على الإطلاق ).

## ٦ - ١٧ تشغيل : عبارة الحساب

تسمح عبارة الحساب COMPUTE بتحديد عديد من العمليات الحسابية مرة واحدة . وعادة ماتنتج عنها برامج هدف أكثر كفاءة عن إجراء الحسابات باستخدام عبارات DIVIDE , MULTIPLY , SUBTRACT, ADD . وعلى هذا .. يجب أن تكتب أى حسابات تشمل أكثر من عملية حسابية واحدة باستخدام COMPUTE .

ولعبارة الحساب شكل واحد فقط ( شكل ٦ - ١٢ ) .

```
COMPUTE identifier-1 [ROUNDED]
    [identifier-2 [ROUNDED]] . . .
        = arithmetic-expression
        [ON SIZE ERROR imperative-statement(s)]
```

شكل (٦ - ١٢)

يشبه التعبير الحسابى arithmatic- expression فى عبارة COMPUTE التعبيرات التى تحدث فى الجبر الأولى، وتتكون من عناصر بيانات عددية، أو ثوابت عددية مخلوطة بمؤثر واحد أو أكثر من المؤثرات الحسابية arithmatiic operators فى الكوبل .

+ الجمع / القسمة
− الطرح ** الأس ( مرفوعاً للقوة )
* الضرب

هذه هى مؤثرات ثنائية binary operators لأنها تتطلب عاملين اثنين two operands تعملان عليهما. يسمح الكوبل كذلك باستخدام إشارة سالب أحادية unary minus sign لنفى القيمة العددية . لايمكن أن تكون العناصر الموجودة فى الناحية اليمنى من علامة التساوى ، التى تدخل الحسابات منقحة ، أما عناصر البيانات الموجودة فى الناحية اليسرى من علامة التساوى، التى تستقبل النتيجة.. فمن الممكن أن تكون منقحة . فى عبارات COMPUTE.. يجب أن يسبق المؤثر الحسابى ويتبعه فراغ، أو يسبقه قوس مغلق، أو يتبعه قوس مفتوح .

**مثـــال ٦ – ٤٩ :**

COMPUTE A = B + C

تكافىء

ADD B C GIVING A

COMPUTE A = B − C

تكافىء

SUBTRACT C FROM B GIVING A

COMPUTE A = B * C

تكافىء

MULTIPLY B BY C GIVING A

COMPUTE A = B / C

تكافىء

DIVIDE B BY C GIVING A

COMPUTE A = B ** C

ليس لها مكافىء فى عبارة أخرى. B**C تمثل B مرفوعا للقوة C أى $B^C$ . فمثلاً B**3 هى تكعيب.

COMPUTE A = − B

( إشارة السالب الأحادية ) تكافىء

SUBTRACT B FROM ZERO GIVING A

يمكن استخدام الأقواس للتحكم فى ترتيب العمليات داخل عبارة COMPUTE : تنفذ العمليات الموجودة فى الاقواس الداخلية جدا اولا ، تليها العمليات الموجودة بين القوسين التاليين ، وهكذا . وفى حالة عدم وجود الأقواس ، يتبع المترجم القواعد التالية :

(١) كل إشارات السالب الفردية تؤدى أولا من اليسار .. لليمين .

(٢) كل عمليات ** تؤدى بعد ذلك من اليسار لليمين .

(٣) كل عمليات * و / تؤدى بعد ذلك من اليسار لليمين .

(٤) كل عمليات + و – تؤدى فى النهاية من اليسار لليمين .

**مثـال ٦ – ٥٠ :**

```
•       COMPUTE A   ROUNDED   = (B + C) * (D + E)
```

بالترتيب : (١) يحسب جمع C , B (٢) يحسب جمع E , D (٣) نتيجة (١) و (٢) تضرب فى بعضهما (٤) يقرب حاصل الضرب (٥) يوضع حاصل الضرب المقرب فى A . ويجب أن تكون العناصر E , D , C , B عددية غير منقحة ، ويمكن أن يكون A منقحا .

```
•       COMPUTE A   ROUNDED
                B
                C               = A + B * C / D
          ON SIZE ERROR
              PERFORM PRINT-ERROR-MESSAGE
```

فى غياب الأقواس ، يحدث بمايلى : (١) يحسب B*C ، (٢) تقسم نتيجة (١) على D ، (٣) تجمع نتيجة (٢) على A ، (٤) توضع نتيجة (٣) فى A ( مع التقريب ) ، (٥) توضع نتيجة (٣) فى B ( مع الحذف ) ، (٦) توضع نتيجة (٣) فى C ( مع الحذف) . إذا حدث خطأ فى الحجم أثناء تخزين النتيجة فى A أو B او C فإن المقطع PRINT- ERROR- MESSAGE ينفذ.

```
•       COMPUTE GROSS-PAY = (40 * RATE) +
                            (1.5 * RATE * (HOURS-WORKED - 40))
```

هذه هى صيغة تقليدية فى حالة العمل وقت أضافى . يدفع المعدل المعتاد على أول 40 ساعة عمل (40*RATE) . ويدفع مرة ونصف من المعدل المعتاد (1.5*RATE) لساعات العمل الأكثر من 40 ساعة (HOURS - WORKED- 40) . وإجمالى الأجر GROSS- PAY هو مجموع القيمتين المدفوعتين .

```
•       COMPUTE NUMBER-EMPLOYEES = NUMBER-EMPLOYEES + 1
```

هذه العبارة التى تبدو غريبة مكافئة لمايلى :

```
ADD 1 TO NUMBER-EMPLOYEES
```

- ```
  COMPUTE QUANTITY-ON-HAND =  QUANTITY-ON-HAND             (100)
                            + QUANTITY-RECEIVED             (10)
                            + QUANTITY-RETURNED-TO-US       (20)
                            - QUANTITY-SHIPPED              (30)
                            - QUANTITY-SCRAPPED             (40)
  ```

تضاف محتويات QUANTITY- RECEIVED , QUANTITY- RETURNED- TO- US إلى القيمة الحالية للعنصر QUANTITY- ON- HAND ، ثم تطرح قيمة QUANTITY-SHIPPED , QUANTITY- SCRAPPED من النتيجة ، وتوضع النتيجة النهائية QUANTITY- ON- HAND مدمرة بذلك القيمة الأصلية التى كانت بها . إذا كان لكل عنصر بيانات القيمة الموجودة بين قوسين أمامه عند البداية ، تصبح المحتويات النهائية للعنصر QUANTITY- ON- HAND هى 60، وتظل بقية عناصر البيانات الأخرى كما هى دون تغيير .

- ```
  COMPUTE AVERAGE-PAY =
      (SALARIED-TOTAL + NONSALARIED-TOTAL) / NUMBER-EMPLOYEES
  ```

يعد استخدام الأقواس ضروريًا هنا ؛ فبدونها .. يقسم NONSALARIED- TOTAL على NUMBER- EMPLOY- EES، وتضاف النتيجة الى SALARIED- TOTAL ، فتنتج عن هذا قيمة خاطئة لمتوسط مايدفع .

- ```
  COMPUTE CLEARANCE-PRICE  ROUNDED  =
      REGULAR-PRICE - .15 * REGULAR-PRICE
  ```

ليس هناك حاجة للأقواس هنا ، حيث أن الضرب يسبق الطرح على أية حال . وفيمايلى صيغة أبسط لنفس الحسابات .

```
COMPUTE CLEARANCE-PRICE  ROUNDED  = .85 * REGULAR-PRICE
```

- ```
  COMPUTE C = (A ** 2 + B ** 2) ** .5
  ```

الرفع إلى القوة 5. هو نفسه مثل أخذ الجذر التربيعى ، وعلى هذا فيحسب الجذر التربيعى لمربع الكمية A مضافا إليه مربع الكمية B . والتمثيل الجبرى لهذا هو مايلى :

$$c = \sqrt{a^2 + b^2}$$

**مثـــال ٦ – ٥١ :**

افرض أن الدفعة الشهرية لقرض تحسب طبقاً لأساس المبلغ، والمبالغ المسددة من الفائدة، والموازنة الجديدة .

```
COMPUTE INTEREST-PAID = OLD-BALANCE * (ANNUAL-INTEREST-RATE / 12)
COMPUTE PRINCIPAL-PAID = PAYMENT-AMOUNT - INTEREST-PAID
COMPUTE NEW-BALANCE = OLD-BALANCE - PRINCIPAL-PAID
```

لاحظ أن معدل الفائدة السنوية يقسم على 12 للحصول على معدل الفائدة الشهرية .

**مثـال ٦ - ٥٢ :**

( استخدام غير صحيح لعبارة COMPUTE) (أ) ثمة خطأ شائع المبتدىء وهو فشله فى ترك فراغ واحد على الأقل، قبل وبعد المؤثر الحسابى ، ولايكون محاطا ب. أو«أو قبل و بعد علامة التساوى» .

```
COMPUTE A = B+C
COMPUTE A = B + C*D
COMPUTE A = (A+B)/(A-B)
```

كلها غير صحيحة ، وذلك بالرغم من أن

```
"COMPUTE A = (A + B)/(A - B)"
```

صحيحة

( ب ) فقدان فراغات محيطة يجعل إشارة السالب تبدو مثل الشرطة .

```
COMPUTE A = B-C
```

يبحث المترجم عن عنصر بيانات اسمه B-C ، ولايفسر هذا كعملية طرح .

( جـ) هناك خطأ يتكرر، وهو الفشل فى الحصول على الحسابات المرجوة بسبب عدم وجود الأقواس . ولايوجد على الأطلاق أى عيب فى استخدام الأقواس – فإذا كان هناك أى شك فيما أذا كان هناك حاجة للأقواس أم لا ، فاستخدمها وأنت متأكد .

```
COMPUTE NEW-BALANCE =    OLD-BALANCE
                       - PAYMENT-AMOUNT
                       + OLD-BALANCE *
                         ANNUAL-INTEREST-RATE / 12
```

تكون صحيحة ، إلا ان مايلى صحيحا كذلك ، له نفس الكفاءة .

```
COMPUTE NEW-BALANCE =
    OLD-BALANCE
  - (PAYMENT-AMOUNT
     - (OLD-BALANCE * (ANNUAL-INTEREST-RATE / 12) ) )
```

لاحظ كيف تقع الأقواس فى أزواج متوازنة ، أى لابد من غلق كل قوس . ويمثل عدم التوازن ( ظهور قوس مفتوح بدون أخر مغلق أو قوس مغلق بدون آخر مفتوح ) خطأ شائع الحدوث .

## ٦ - ١٨ الكفاءة فى الحسابات الرياضية

بصفة عامة.. يمكن الحصول على برنامج هدف أكثر كفاءة عندما :

( ١ ) تكون لعناصر البيانات المشمولة فى الحسابات PICTURE محددة فيها العلامة العشرية .

( ٢ ) يكون لعناصر البيانات المشمولة فى الحسابات نفس الاستخدام USAGE .

( ٣ ) لعناصر البيانات المستخدمة فى الحسابات استخدام من نوع COMP أو COMP-3 .

( ٤ ) تكون لحقول COMP-3 ( فى كوبل IBM OS/VS ) عدد أرقام فردى ( وبصفة خاصة عند استخدام ON SIZE ERROR) .

( ٥ ) يكون أحد الحقول فى سجل منطقى، استخدام DISPLAY ، ويكون مشمولا فى عديد من الحسابات ؛ فينقل الحقل إلى منطقة مخزن عمل، تكون لها استخدام COMP أو COMP-3 وتستخدم نسخة مخزن العمل هذه فى الحسابات . (أى يحدث تحويل واحد للاستخدام بدلا من عديد من التحويلات) .

( ٦ ) تستخدم S فى PICTURE العددية المشمولة فى الحسابات ( إلا إذا ما كان المطلوب هو القيمة المطلقة ) .

( ٧ ) يتم تجنب مواصفة ON SIZE ERROR بالاختيار المناسب للصور PICTURES .

## ٦ - ١٩ نمطيات كتابة شفرة اضافية

## لجزء الاجراءات

( ارجع إلى القسم ٦ – ١ )

( ١١ ) استخدم STOP RUN مرة واحدة فقط فى البرنامج .

( ١٢ ) العبارات الطويلة تجزأ عند بدء الجزء ثم ترحل فى السطر التالى .

( ١٣ ) ضع الأجزاء مثل ON SIZE ERROR , AT END- OF- PAGE , AT END مرحلة على السطور الخاصة بها .

( ١٤ ) أضبط العبارات أو أجزاء العبارات المتشابهة رأسيا مع بعضها البعض .

( ١٥ ) استخدم ثابتًا أذا كنت متأكدا أنه لن يتغير على الإطلاق أثناء فترة استخدام البرنامج .

**مثـال ٦ – ٥٣ :**

- MOVE ZERO TO NUMBER-OF-EMPLOYEES

كما توضع NUMBEER- OF- EMPOIYEES مساوية صفرا فى بداية تنفيذ البرنامج دائما.. فإن استخدام الثابت الاستعارى ZERO يكون مناسبا .

- MULTIPLY REGULAR-PRICE BY .08 GIVING EMPLOYEE-DISCOUNT

يمكن أن يتغير معدل خصم العميل من 08. إلى قيمة أخرى ، ويجب ألا يستخدم ثابت هنا . بدلا من ذلك.. يجب أن يعرف المبرمج عنصر بيانات فى مخزن العمل على النحو التالى :

```
WORKING-STORAGE SECTION.

01  PROGRAM-CONSTANTS.
    05  EMPLOYEE-DISCOUNT-RATE      PIC SV99
                                    VALUE +.08.
    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

ويجب أن يستبدل MULTIPLY السابقة بما يلى :

```
MULTIPLY REGULAR-PRICE BY EMPLOYEE-DISCOUNT-RATE GIVING
        EMPLOYEE-DISCOUNT
```

والآن ، إذا وجب تغيير معدل خصم العميل.. فلن يكون هناك حاجة الا إلى تغيير VALUE واحدة فقط ؛ بدلا من تغيير الثابت نفسه فى كل موقع يظهر فيه جزء الإجراءات .

## اسئلة مراجعة

٦ – ١ ماالغرض من جزء الإجراءات ؟

٦ – ٢ أشرح هيكل جزء الإجراءات بتعريف : القسم، والمقطع، والجملة، والعبارة، والفعل .

٦ – ٣ ناقش نمطيات كتابة الشفرة لكتابة جزء الإجراءات .

٦ – ٤ ماوظيفة عبارة الفتح OPEN ؟

٦ – ٥ وضح كل حالة من حالات عبارة الفتح : INPUT , OUTPUT , I/O , EXTEND .

٦ – ٦ ناقش استخدام عبارة الفتح فى فتح كل الملفات فى فتح كل ملف على حدة .

٦ – ٧ ماذا يحدث إذا ماأشار البرنامج إلى منطقة سجل منطقى للف قبل فتح الملف، أو بعد إغلاق الملف ؟

٦ – ٨ ماوظيفة عبارة الإغلاق CLOSE ؟

( ٦ – ٩ قارن بين إغلاق كل الملفات بعبارة إغلاق واحدة، وبين إغلاق كل ملف بعبارة إغلاق مستقلة به .

٦ – ١٠ ناقش كيف يمكن فتح وإغلاق ملف عديد من المرات بواسطة نفس البرنامج .

٦ – ١١ ما وظيفة WITH LOCK فى عبارة CLOSE ؟

٦ – ١٢ ما الشريط متعدد الملفات ؟

٦ – ١٣ وضح استخدام STANDARD LABELS مع الشريط متعدد الملفات .

٦ – ١٤ ما الملف متعدد الحجوم ؟

٦ – ١٥ ماذا يعنى عمل مفتاح للحجم volume switching ؟

٦ – ١٦ ما الغرض من عبارة القراءة READ ؟

٦ – ١٧ وضح الغرض من جزء AT END فى عبارة القراءة.

٦ – ١٨ ما شرط نهاية الملف ؟ ومتى يتحقق هذا الشرط ؟

٦ –١٩ لماذا يكون من الضرورى عدم تشغيل سجل منطقى بعد حدوث نهاية الملف ؟

٦ – ٢٠ لماذا يختار بعض المبرمجين عمل تشغيل كل السجلات المنطقية، باستخدام نسخ السجلات المنطقية فى مخزن العمل ؟

٦ – ٢١ وضح استخدام READ... INTO .

٦ – ٢٢ ماذا يعنى شعار « اقرأ ملفًا واكتب سجلا » ؟

٦ – ٢٣ ماالغرض من عبارة الكتابة WRITE ؟

٦ – ٢٤ ماذا يوجد فى منطقة سجل منطقى للف فور إتمام عملية كتابة WRITE بنجاح ؟

٦ – ٢٥ ما بعض مميزات بناء سجلات منطقية فى منطقة مخزن عمل، ثم استخدام WRITE... FROM – بعد ذلك – فى نلقها إلى ذاكرة احتياطية للمخرجات ؟

٦ – ٢٦ إذا لم يمكن استخدام WRITE... FROM , READ... INTO مع سجلات متغيرة الأطوال .. كيف يستطيع البرنامج تشغيل مثل هذه السجلات فى مناطق مخزن العمل ؟

٦ – ٢٧ ناقش الطرق المختلفة للتحكم فى فراغات الطابع للملفات التى يتم إخراجها عن طريق الطابع .

٦ – ٢٨ ماذا يحدث إذا لم تشمل الكتابة فى ملف على الطابع جزءًا لمسافات الطابع ؟

٦ – ٢٩ اذكر بعض الأمثلة لمواقف، يجب استخدام عنصر بيانات فيها لتحديد عدد الأسطر فى جزء BEFORE/AFTER ADVANCING من عبارة الكتابة .

٦ – ٣٠ ما المعانى الممكنة لـ BEFORE/AFTER ADVANCING PAGE ؟

٦ – ٣١ وضح كيف يمكن استخدام قنوات تحكم العربة المحددة، فى مقطع الأسماء الخاصة، فى التحكم فى فراغات الطابع WRITE .

٦ – ٣٢ وضح مناطق صفحة منطقية التى تعرف بجزء LINAGE .

٦ – ٣٣ اذكر الشرطين الذى ينفذ تحتهما مقطع AT END- OF- PAGE ؛ كجزء من عبارة الكتابة .

٦ – ٣٤ ماذا يمكن أن يحدث إذا لم تتسق الأجزاء التالية بطريقة مناسبة: LINAGE ، وجزء BEFORE/ AFTER ، وجزء AT END- OF- PAGE ؟

٦ – ٣٥ ما السريان الزائد التلقائى للصفحة ؟

٦ – ٣٦ عرف العبارة الشرطية والعبارة الأمرية .

٦ – ٣٧ متى تكون عبارة القراءة READ شرطية ؟ ومتى تكون أمرية ؟

٦ – ٣٨ متى تكون عبارة الكتابة WRITE شرطية ؟ ومتى تكون أمرية ؟

٦ – ٣٩ مانوع معلومات نظام التشغيل الذى يمكن الحصول عليه بعبارة ACCEPT ؟ وكيف يمكن استخدام هذه المعلومات ؟

٦ – ٤٠ اذكر بعض الأمثلة التى يمكن استخدام ACCEPT فيها لإدخال معلومات من النهاية الطرفية المتاحة لمشغل الكمبيوتر.

٦ – ٤١ متى يجب ألا تستخدم ACCEPT بدلاً من إعداد ملف معتاد باستخدام CLOSE , REEAD , OPEN , FD , SELEECT ؟

٦ – ٤٢ ماالمقصود ببطاقات التحكم ؟

٦ – ٤٣ كيف يمكن استخدام عبارة DISPLAY فى تصحيح برنامج ؟

٦ – ٤٤ متى يمكن استخدام DISPLAY لرسائل الخطأ ( بدلا من ملف المخرجات المعتاد المطبوع باستخدام CLOSE, WRITE, OPEN, FD SELECT) ؟

٦ – ٤٥ اذكر بعض الأمثلة التى يمكن استخدام DISPLAY فيها؛ للاتصال مع مشغل الكمبيوتر عبر نهايته الطرفية .

٦ – ٤٦ بالاضافة الى نقل بيانات.. ما الوظائف الأخرى التى تؤديها عبارة MOVE ؟

٦ – ٤٧ عرف : (أ) الحقل الراسل . (ب) الحقل المستقبل (ج) التحويل (د) الملء بفراغات أو أصفار (هـ) الحذف (و) التنقيح .

٦ – ٤٨ ماهى قواعد الحذف والملء المستخدمة (أ) مع البيانات العددية (ب) مع البيانات غير العددية عندما يحدث نقل لها ؟

٦ – ٤٩ ماذا يحدث فى الكوبل عندما تنقل مجموعة عناصر ؟ اذكر بعض الأخطاء شائعة الحدوث عند نقل مجموعة عناصر ؟

٦ – ٥٠ متى يكون تنفيذ نقل مجموعة عناصر آمنا ؟

٦ – ٥١ صف خواص خليط العناصر الفردية الذى لايكون صحيحاً فى عبارة النقل و اذكر عدة امثلة لنقل فردى غير صحيح.

٦ – ٥٢ ما وظيفة عبارة STOP RUN ؟

٦ – ٥٣ لماذا يجب أن تغلق كل الملفات قبل ايقاف تنفيذ البرنامج ؟

٦ – ٥٤ ماذا يحدث عند تنفيذ العبارة "STOP "GIVE ME A PEN ؟

٦ – ٥٥ ما الغرض من عبارة PERFORM ؟

٦ – ٥٦ ميز بين الترتيب الواقعى لعبارات جزء الإجراءات وترتيبها المنطقى ؟

٦ – ٥٧ ما دورة loop البرنامج ؟

٦ – ٥٨ وضح بالتفصيل كيفية عمل عبارة PERFORM بسيطة .

٦ – ٥٩ وضح بالتفصيل كيفية عمل عبارة PERFORM... UNTIL .

٦ – ٦٠ ماذا يحدث إذا تحقق شرط PERFORM ... UNTIL فجأة فى منتصف تنفيذ مقطع PERFORM ؟

٦ – ٦١ ما برنامج التنقيح ؟

٦ – ٦٢ ما النسخة الدائمة ؟

٦ – ٦٣ ما الغرض من عبارة ADD ؟

٦ – ٦٤ ميز بين تأثيرات الحذف التقليدى وجزء ROUNDED عند الحصول على نتائج كسرية أثناء حسابات الكمبيوتر .

٦ – ٦٥ مامعنى السريان الزائد ؟

٦ – ٦٦ وضح استخدام جزء ON SIZE ERRO ، فى العبارة الحسابية .

٦ –٦٧ ماذا يمكن عمله لجعل مقطع ON SIZE ERROR غير ضرورى فى البرنامج ؟

٦ – ٦٨ اذكر خطوطا إرشادية لكتابة مقطع ON SIZE ERROR .

٦ – ٦٩ أين يمكن استخدام عناصر عددية منقحة فى العبارات الحسابية ؟

٦ – ٧٠ وضح الغرض من عبارة MULTIPLY . كيف يختلف تكوينها عن تكون عبارة ADD ، وعبارة SUBTRACT؟

٦ – ٧١ ما الغرض من عبارة DIVIDE ؟

٦ – ٧٢ عرف : (أ) خارج القسمة (ب) الباقى .

٦ – ٧٣ ماذا يحدث فى الكوبل ، إذا حاولت القسمة على صفر ؟

٦ – ٧٤ اذكر بعض ميزات استخدام عبارة COMPUTE ؟

٦ – ٧٥ اذكر المؤثرات الخمس الثنائية المستخدمة فى عبارة COMPUTE .

٦ – ٧٦ ماذا تعنى إشارة السالب الأحادية ؟

٦ – ٧٧ ما دور الأقواس فى عبارة COMPUTE ؟

٦ – ٧٨ ما الترتيب التقليدى لتنفيذ العمليات الحسابية إذا لم تستخدم الأقواس ؟

٦ – ٧٩ اذكر بعض الأخطاء المعتادة فى استخدام COMPUTE ؟

٦ - ٨٠ ناقش مايمكن عمله فى الكوبل لجعل العمليات الحسابية ذات كفاءة بقدر الإمكان .

٦ - ٨١ متى يمكن أن تستخدم الثوابت بأمان فى البرنامج ؟

٦ - ٨٢ ما الطريقة التى يمكن استخدامها للإحلال محل الثوابت في جزء الإجراءات ؟ ومتى يجب استخدامها ؟ لماذا تجعل هذه الطريقة البرنامج أسهل فى تعريفه فيما بعد ؟

## مسائل محلولة

٦ - ٨٣ وضح العبارة : «تشمل معظم خوارزميات الأعمال تكرارا لدورة مدخلات - تشغيل - مخرجات - حتى نهاية الملف»

الخوارزم algorithm عبارة عن مجموعة ليست غامضة من الخطوات التفصيلية، لعمل نشاط معين أو حل مشكلة معينة. تشمل مشاكل الأعمال التقليدية تشغيلا لكل السجلات المنطقية الموجودة في ملف المدخلات، وهذه المشاكل مبنية على ذلك حول دورة رئيسية main loop يحدث فيها: (١) يكون السجل التالى مدخلات (٢) يتم تشغيل لسجل المدخلات المنطقى الذى دخل توا (٣) اذا كان ممكنا ، تخرج نتائج تشغيل السجل المنطقى . وتتكرر الخطوات الثلاث من (١) الى (٣) حتى تصل لنهاية الملف أثناء محاولة تنفيذ الخطوة رقم (١) .

٦ - ٨٤ يقرأ أحد البرامج ملف مدخلات، به سجلات تاريخ المبيعات، ويضيفها إلى ملف آخر موجود . وضح كيف يفتح كل من الملفين.

```
OPEN    INPUT     SALES-HISTORY-TRANSACTIONS
        EXTEND    SALES-HISTORY-MASTER
```

جزء EXTEND ، يسمح بإضافة سجلات جديدة الى نهاية ملف تتابعى موجود فعلاً .

٦ - ٨٥ افرض فى المسألة السابقة.. أنه من المطلوب حذف كل المحتويات السابقة للملف واستبدالها بسجلات جديدة كلية . اكتب عبارة الفتح اللازمة لذلك .

```
OPEN    INPUT     SALES-HISTORY-TRANSACTIONS
        OUTPUT    SALES-HISTORY-MASTER
```

عندما يفتح ملف تتابعى موجود فعلا كملف مخرجات ، تحذف كل السجلات المنطقية السابقة منه، وينتج صيغة جديدة تماماً للملف الناتج من كتابة WRITING سجلات منطقية فيه.

٦ - ٨٦ حدد الخطأ فى جزء الإجراءات التالى :

```
FD  INPUT-FILE . . .
01  INPUT-RECORD . . .

FD  REPORT-FILE . . .
01  REPORT-LINE . . .
```

```
PROCEDURE DIVISION.

    MOVE SPACES TO INPUT-RECORD
    WRITE REPORT-LINE FROM WS-HEADING-LINE
        AFTER ADVANCING PAGE
    OPEN  INPUT    INPUT-FILE
          OUTPUT   REPORT-FILE
```

يوضح هنا خطآن شائعان، هما :

(١) تتصل MOVE SPACES TO INPUT- RECORD بمنطقة سجل منطقى للملف قبل فتح الملف، ويتسبب ذلك فى حدوث أخطاء جسيمة دائما ، حيث لاتتاح ذاكرات الملف الاحتياطية بعد للبرنامج .

(٢) تتصل WRITE REPORT- LINE FROM بمنطقة سجل منطقى قبل فتح الملف ( حيث إن WRITE... FROM تكافيء عبارة MOVE يتبعها عبارة WRITE ) وأسوأ من ذلك .. يتم فى محاولة إخراج سجل قبل فتح الملف، حيث يتسبب هذا الخطأ الأخير فى إنهاء البرنامج نهاية غير طبيعية .

٦ – ٨٧ تنقل كل من CLOSE , WRITE , READ, OPEN مفتاح حالة مكون من 2 بايت فى منطقة FILE STATUS للملف (بافتراض أن FILE STATUS معرفة فى الملف).

ما قيمة مفتاح الحالة التي تحدد تنفيذا ناجحا لأى من هذه العبارات ؟

فى كوبل IBM SO/VS .. دائما مايحدد 00 اتماما ناجحا لعبارة مدخلات أو مخرجات . يمكن استخدام PIC 99 أو PIC XX لمفتاح الحالة .

٦ – ٨٨ حدد الخطأ فيمايلى :

```
05  FILE-STATUS-AREA      PIC XX.
. . . . . . . . . . . . . . . . . . . . . . . . . . . .
OPEN   INPUT   SAMPLE-FILE
IF FILE-STATUS-AREA  EQUAL 00 . . .
```

منطقة مفتاح الحالة المعرفة بالصورة PIC XX تكون غير عددية ، وعلى هذا .. يجب استخدام ثابت غير عددى :

```
IF FILE-STATUS-AREA  EQUAL "00" . . .
```

٦ – ٨٩ ماذا يمكن أن يتسبب فى فشل عبارة CLOSE ( ويضع رمزاً غير صفرى فى منطقة FILE STATUS ) ؟

إذا كانت للملف عناوين نمطية .. فيمكن أن يتسبب فشل نظم المكونات فى أخطاء مدخلات ومخرجات غير قابلة للتصليح، وذلك أثناء تشغيل العنوان . وهناك إمكانية أخرى وهى أنه قد لايفتح الملف فتح بنجاح ، وفى هذه الحالة.. لن يغلق بنجاح.

٦ – ٩٠ بين كيف تغلق ٣ ملفات مقللا: (أ) وقت التنفيذ (ب) عدد البايت من الذاكرة .

(أ)

```
CLOSE INVENTORY-TRANSACTION-FILE
      INVENTORY-MASTER-FILE
      INVENTORY-UPDATE-FILE
```

عندما نغلق عدة ملفات بعبارة CLOSE واحدة .. يقل وقت التنفيذ ( الا أنه هناك حاجة إلى مزيد من ذاكرة الكمبيوتر).

(ب)

```
CLOSE INVENTORY-TRANSACTION-FILE
CLOSE INVENTORY-MASTER-FILE
CLOSE INVENTORY-UPDATE-FILE
```

يقلل إغلاق كل ملف على حدة من استخدام الذاكرة، إلا أنه يستغرق وقتا أطول .

٦ – ٩١ افرض أن هناك برنامجا يقوم بتشغيل ملفين على نفس الشريط ( شريط متعدد الملفات ) . يفتح البرنامج الملف الأول ويشغله، ثم يغلق ويفتح الملف الثانى . اكتب عبارتى الفتح وعبارة إغلاق واحدة لهذا البرنامج .

```
OPEN    INPUT   FIRST-FILE-ON-TAPE
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
CLOSE FIRST-FILE-ON-TAPE   WITH NO REWIND
OPEN    INPUT   SECOND-FILE-ON-TAPE
```

٦ – ٩٢ عندما يقوم أحد البرامج بتشغيل ملف متعدد الأحجام تتابعيا.. فعادة مايعامل نظام التشغيل تحويل أحجام تلقائيا automatic volume switching . وضح كيفية عمل ذلك (أ) لملف مدخلات (ب) لملف مخرجات .

(أ) يبدأ البرنامج قراءة سجلات منطقية من أول حجم من أحجام الملف المتعدد . إذا كان هناك 500 سجل منطقى على أول حجم للملف .. فإن تنفيذ عبارة READ رقم 501 يتسبب فى تحويل الحجم ، الذى يقوم فيه نظام التشغيل بمايلى: (١) التأكد من أن الحجم الثانى من الملف موضوع بطريقة مناسبة (٢) تشغيل أى سجلات عناوين نمطية للحجم الثانى (٣) جعل أول سجل منطقى من الحجم الثانى للملف متاحا للبرنامج .

(ب) عندما يفتح ملف مكون من عدة أحجام كمخرجات ، فيوضع الملف عند بداية أول حجم . عند ذلك يبدأ البرنامج كتابة سجلات منطقية فى الحجم الأول . عندما ينفذ البرنامج عبارة WRITE ، ولايجد مكانا متبقيا للتسجيل على الحجم الأول ؛ فيقوم نظام التشغيل بعمل مايلى تلقائيا: (١) إنهاء تشغيل أى عناوين نمطية للحجم الأول (٢) بدء تشغيل العناوين النمطية للحجم الثانى (٣) كتابة سجل منطقى فى بداية الحجم الثانى .

٦ – ٩٣ اربط عبارة CLOSE... UNIT بتغيير الحجم تلقائيا .

لقد رأينا فى المسألة السابقة.. كيف يبدأ نظام التشغيل فى تغيير الحجم تلقائيا، عندما يصل إلى نهاية الحجم أثناء

التشغيل المعتاد للملف . ويمكن استخدام عبارة CLOSE... UNIT فى إجبار تغيير الحجم عند اى نقطة اختيارية فى البرنامج . ويمكن أن يكون هذا مفيدا عندما يتم إنتاج ملف متعدد الحجوم ؛ لانه يسمح للبرنامج بالتحكم مباشرة فى عدد السجلات المنطقية التى تكتب في كل حجم .

٦ – ٩٤ حدد الخطأ فيمايلى :

```
OPEN  INPUT  SAMPLE-FILE
READ SAMPLE-FILE
    AT END
        MOVE "T" TO END-OF-FILE-SW
MOVE SAMPLE-RECORD TO OUTPUT-AREA
WRITE OUTPUT-RECORD
    FROM OUTPUT-AREA.
```

هناك أخطاء عديدة : (١) لاتوجد نقطة تحدد نهاية جزء AT END . (٢) تنفذ WRITE OUTPUT- RECORD بدون فتح ملف مخرجات . (٣) يفشل البرنامج فى أختبار نهاية الملف قبل محاولته MOVE SAMPLE- RECORD TO OUTPUT- AREA . إذا حدثت نهاية الملف ، فلن يوجد SAMPLE- RECORD لتشغيله (ويدخل البرنامج قيمة نفايا فى التشغيل).

٦ – ٩٥ الهدف من جزء الإجراءات التالى هو نسخ ملف به سجلات على بطاقات من 80 عموداً فى ملف قرص . صحح الأخطاء

```
PROCEDURE DIVISION.

    OPEN  INPUT  CARD-FILE
    MOVE "NO" TO END-OF-FILE
    PERFORM COPY-A-RECORD
        UNTIL END-OF-FILE EQUAL "YES"
    CLOSE DISK-FILE

COPY-A-RECORD.

    READ CARD-FILE
        AT END  MOVE "YES" TO END-OF-FILE
    MOVE CARD-RECORD TO DISK-RECORD
    WRITE DISK-RECORD
```

الأخطاء بترتيب حدوثها، هى : (١) لم يفتح DISK- FILE . (٢) لم يغلق CARD- FILE . (٣) لاتوجد STOP RUN . (٤) لاتوجد نقطة بعد جزء AT END (٥) لايوجد اختبار لنهاية الملف، قبل محاولة نقل MOVE ، وكتابة WRITE سجل بطاقة . الصيغة المصححة هى كما يلى :

```
PROCEDURE DIVISION.

       OPEN  INPUT    CARD-FILE
             OUTPUT   DISK-FILE
       MOVE "NO" TO END-OF-FILE
       PERFORM COPY-A-RECORD
          UNTIL END-OF-FILE EQUAL "YES"
       CLOSE CARD-FILE
             DISK-FILE
       STOP RUN
       .
COPY-A-RECORD.

       READ CARD-FILE
          AT END  MOVE "YES" TO END-OF-FILE
       .
       IF END-OF-FILE EQUAL "NO"
          MOVE CARD-RECORD TO DISK-RECORD
          WRITE DISK-RECORD
```

٦ – ٩٦ بافتراض أن كل الملفات فتحت ... إلخ ، حدد الخطأ فيما يلي:

```
    WRITE SAMPLE-OUTPUT-RECORD
       FROM WS-OUTPUT-RECORD-AREA

    COMMENT -- DISPLAY RECORD JUST WRITTEN:
*      FOR DEBUGGING

    DISPLAY SAMPLE-OUTPUT-RECORD
```

بعد تنفيذ عبارة WRITE .. لايشير وصف السجل المسمى SAMPLE- OUTPUT- RECORD للسجل المنطقى الذى كتب توا . وبدلا من ذلك .. فهو يشير الى موقع سجل منطقى جديد فى ذاكرة احتياطية الملف ؛ يمكن استخدامها في عمل السجل المنطقى التالى المراد كتابته . وتعرض عبارة DISPLAY نفايا ، ولاتعرض السجل الذى كتب توا .

٦ – ٩٧ صحح الخطأ المذكور فى المسألة السابقة ؛ حيث ان جزء WRITE... FROM قد استخدم ، فيكون التصحيح التعويض بـ :

```
DISPLAY WS-OUTPUT-RECORD-AREA
```

(بعد التنفيذ لعبارة WRITE .. فإن WS- OUTPUT- RECORD- AREA يظل محتويا على السجل الذي كتب توا) .

إذا كانت فى المسألة السابقة عبارة WRITE بسيطة ؛ أى فيها :

```
WRITE SAMPLE-OUTPUT-RECORD
DISPLAY SAMPLE-OUTPUT-RECORD
```

فيكون التصحيح سهلا، ويأخذ الشكل التالى :

```
DISPLAY SAMPLE-OUTPUT-RECORD
WRITE SAMPLE-OUTPUT-RECORD
```

٦ – ٩٨ حدد الخطأ فيما يلى :

```
FD  INPUT-FILE
    RECORD CONTAINS 50 TO 200 CHARACTERS
. . . . . . . . . . . . . . . . . . . . . . . . .
OPEN    INPUT    INPUT-FILE
READ INPUT-FILE
    INTO WS-INPUT-RECORD-AREA
    AT END...
```

لاتستخدم READ... INTO مع سجلات متغيرة الطول ( انظر ملحق حـ ؛ لمعرفة الموقف فى كوبل 1985 النمطى ).

٦ – ٩٩ بين كيف يصحح الخطأ الموجود فى المسألة السابقة .

```
FD  INPUT-FILE
    RECORD CONTAINS 50 TO 200 CHARACTERS
. . . . . . . . . . . . . . . . . . . . . . . . .
OPEN    INPUT    INPUT-FILE
READ INPUT-FILE
    AT END...
MOVE INPUT-RECORD TO WS-INPUT-RECORD-AREA
```

استخدم عبارة READ تتبعها عبارة MOVE ، بدلا من READ... INTO ، وذلك مع السجلات متغيرة الطول . ومع نفس السجلات.. استخدمت كذلك عبارة MOVE .. يتبعها عبارة WRITE بدلا من استخدام WRITE... FROM .

٦ – ١٠٠ حدد الخطأ فيما يلى :

```
READ SAMPLE-INPUT-FILE
    AT END
        MOVE "YES" TO END-OF-FILE
        WRITE REPORT-LINE
            FROM WS-TOTAL-LINE
            AT END-OF-PAGE
                ADD 1 TO NUMBER-OF-PAGES
```

يجب أن تكون كل العبارات فى جزء AT END أمرية ، وهنا WRITE... AT END- OF- PAGE هى شرطية ، وينتج عن ذلك خطأ تكوينى .

٦ – ١٠١ صحح الخطأ الموجود فى المسألة السابقة .

اذا كان ضروريا تنفيذ عبارة شرطية، عندما تكون هناك قواعد تطلب وجود عبارة أمرية ... ضع العبارة الشرطية فى مقطع، ثم استخدم عبارة PERFORM ( وهى أمرية ) فى تنفيذ المقطع .

```
READ SAMPLE-INPUT-FILE
    AT END
        PERFORM AT-END-ROUTINE

............................
AT-END-ROUTINE.

    MOVE "YES" TO END-OF-FILE
    WRITE REPORT-LINE
        FROM WS-TOTAL-LINE
        AT END OF PAGE
            ADD 1 TO NUMBER-OF-PAGES
```

٦ – ١٠٢ مطلوب إنتاج قائمة من مجموعة بطاقات ؛ حيث تطبع نسخة طبق الأصل من كل بطاقة ( الثمانين عموداً لها ) على الورق. يجب أن تعيد كل صفحة منطقية إنتاج 10 بطاقات بالضبط ، ويجب أن يكون فيها سطر عنوان، يحتوى على عنوان ورقم الصفحة والتاريخ فى صورة mmyydd ، وفى صورة رقم اليوم من السنة، ووقت تنفيذ البرنامج على الكمبيوتر، كما يجب أن يكون للصفحة نهاية، بها الرسالة SO FAR- CARDS WERE PRINTED . علق على كيفية تحقيق البرنامج الموجود فى شكل (٦ – ١٣) لهذا النشاط .

```
       IDENTIFICATION DIVISION.
       PROGRAM-ID. CARDLIST.
       AUTHOR.  LARRY NEWCOMER.
       INSTALLATION.  PENN STATE UNIVERSITY--YORK CAMPUS.
       DATE-WRITTEN.  MAY 1983.
       DATE-COMPILED. MAY  9,1983.
       SECURITY.  NONE.
      *    CARDLIST PRODUCES AN 80/80 LISTING OF CARD CONTENTS.
      *    ITS PURPOSE HERE IS TO ILLUSTRATE VARIOUS FORMS OF
      *    WRITE ... BEFORE/AFTER ADVANCING ... WITH THE LINAGE
      *    CLAUSE.
       ENVIRONMENT DIVISION.
       CONFIGURATION SECTION.
       SOURCE-COMPUTER.  IBM-370.
       OBJECT-COMPUTER.  IBM-370.
       INPUT-OUTPUT SECTION.
       FILE-CONTROL.
           SELECT CARD-FILE             ASSIGN TO CARDS.
           SELECT PRINT-FILE            ASSIGN TO PRINTER.
       DATA DIVISION.
```

```
       FILE SECTION.
       FD  CARD-FILE
           RECORD CONTAINS 80 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .
       01  CARD-INPUT                      PIC X(80).

      *    ----------------------------------------
       FD  PRINT-FILE
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED
           LINAGE IS 13
               WITH FOOTING AT 11
               LINES AT TOP 2
               LINES AT BOTTOM 2


       01  PRINT-LINE                      PIC X(132).
      *    ----------------------------------------

       WORKING-STORAGE SECTION.

      *    ----------------------------------------
       01  WS-DATE-AND-TIME-AREAS.
           05  WS-REGULAR-DATE.
               10  WS-YY                   PIC 99.
               10  WS-MM                   PIC 99.
               10  WS-DD                   PIC 99.
      *    ----------------------------------------

       01  PROGRAM-SWITCHES-AND-COUNTERS.
           05  END-OF-CARDS-SWITCH         PIC X(3).
           05  WS-PAGE-NUMBER              PIC S9(3)       COMP-3.
           05  WS-CARD-COUNT               PIC S9(3)       COMP-3.
       01  WS-LINE-AREA.
           05  WS-CARD-AREA                PIC X(80).
           05  FILLER                      PIC X(52)       VALUE SPACES.

      *    -----------------------------------------------------------
       01  WS-HEADING-AREA.
           05  FILLER                      PIC X(6)        VALUE "DATE ".
           05  HEADING-MM                  PIC Z9.
           05  FILLER                      PIC X           VALUE "/".
           05  HEADING-DD                  PIC 99.
           05  FILLER                      PIC X           VALUE "/".
           05  HEADING-YY                  PIC 99.
           05  FILLER                      PIC X(9)        VALUE ", JULIAN ".
           05  HEADING-JULIAN              PIC Z9B999.
           05  FILLER                      PIC X(7)        VALUE ", TIME ".
           05  HEADING-TIME                PIC 99B99B99B99.
           05  FILLER                      PIC X(6)        VALUE " PAGE ".
           05  WS-HEADING-PAGE-NUMBER      PIC ZZ9.
           05  FILLER                      PIC X(76)       VALUE SPACES.
      *    -----------------------------------------------------------

       01  WS-FOOTING-AREA.
           05  FILLER                      PIC X(7)        VALUE "SO FAR ".
           05  WS-FOOTING-COUNT            PIC ZZ9.
           05  FILLER                      PIC X(122)
                                           VALUE " CARDS WERE PRINTED".

       PROCEDURE DIVISION.

           MOVE "NO " TO END-OF-CARDS-SWITCH
           MOVE ZERO  TO WS-PAGE-NUMBER
                         WS-CARD-COUNT
```

```

           ACCEPT WS-REGULAR-DATE       FROM DATE
           ACCEPT HEADING-JULIAN        FROM DAY
           ACCEPT HEADING-TIME          FROM TIME

           MOVE WS-MM                   TO HEADING-MM
           MOVE WS-DD                   TO HEADING-DD
           MOVE WS-YY                   TO HEADING-YY

           OPEN     INPUT               CARD-FILE
                    OUTPUT              PRINT-FILE

           PERFORM INPUT-A-RECORD

           PERFORM PRODUCE-PAGE-HEADING

           PERFORM PRODUCE-80-80-LISTING
               UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"

           CLOSE    CARD-FILE
                    PRINT-FILE
           STOP RUN
           .

       INPUT-A-RECORD.

           READ CARD-FILE RECORD
               INTO WS-CARD-AREA
               AT END
                   MOVE "YES" TO END-OF-CARDS-SWITCH
           .

       PRODUCE-80-80-LISTING.

           ADD 1 TO WS-CARD-COUNT

           WRITE PRINT-LINE
               FROM WS-LINE-AREA
               AFTER  ADVANCING 1 LINE
               AT END-OF-PAGE
                   PERFORM PRODUCE-PAGE-FOOTING
                   PERFORM PRODUCE-PAGE-HEADING
           .

           PERFORM INPUT-A-RECORD
           .

       PRODUCE-PAGE-HEADING.

           ADD 1 TO WS-PAGE-NUMBER
           MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE-NUMBER
           WRITE PRINT-LINE
               FROM WS-HEADING-AREA
               AFTER ADVANCING PAGE
           .

       PRODUCE-PAGE-FOOTING.

           MOVE WS-CARD-COUNT TO WS-FOOTING-COUNT

           WRITE PRINT-LINE
               FROM WS-FOOTING-AREA
               AFTER ADVANCING 2 LINES
           .
```

شكل (٦ - ١٣)

(١) كما يبدأ البرنامج بقراءة أول بطاقة ( بعد فتح الملفات فورا ) .. فلابد أن يبدأ بطباعة عنوان فى بداية الصفحة الاولى . ويسمح وضع مقطع العنوان فى هذا المقطع الخاص بتنفيذه فى بداية جزء الإجراءات وتنفيذه مرة أخرى ، كلما كانت هناك حاجة لذلك، فى مقطع AT END- OF- PAGE من جزء الاجراءات . وحيث إن الطباعة تبدأ فى بداية اول صفحة منطقية.. فلا تكتشف END- OF - PAGE ، فى أول كتابة WRITE للملف . ويؤكد السطر الخامس من شكل (٦ – ١٣) أن أول صفحة سيكون بها عنوان .

(٢) يختبر النظام بالنسبة للسريان الزائد للنهاية footing قبل اختباره للسريان الزائد للصفحة . وعندما تطبع البطاقة العاشرة ( فى السطر الحادى عشر بسبب طباعة سطر عنوان ) يتميز السريان الزائد للنهاية ، وينفذ مقطع AT END- OF- PAGE ؛ متسببا فى طباعة سطر النهاية ، بعد ذلك .. يطبع سطر فى بداية صفحة منطقية جديدة . وعلى هذا ، عندما تطبع البطاقة الحادية عشرة ، يوضع الورق فعلا عند بداية الصفحة ؛ ولايحدث سريان للصفحة . ونرى ان استخدام منطقة نهاية footing فى نهاية الصفحة يمنع حدوث سريان زائد للصفحة؛ عن طريق كتابة سطر عنوان فى بداية الصفحة التالية.

(٣) تتحدد قيمة LINAGE (السطر الثانى والثلاثون من شكل ٦ – ١٣) كمايلى : حيث إن العنوان يتطلب سطرا واحدا، ونحن نريد عشرة سطور ( تناظر عشرة بطاقات ) فى الصفحة ، إذ يجب أن تكون قيمة LINAGE مساوية 11 على الأقل ( بدون نهاية footing الصفحة ) . فإذا كان المطلوب طباعة النهاية بعد تقديم سطرين AFTER ADVANCING 2 LINES .. فإن السطرين الإضافيين يعتبران من حجم جسم الصفحة 13 . ويبين شكل ٦ – ١٤ عينة لصفحة مطبوعة .

```
DATE    5/09/83, JULIAN 83 129, TIME 13 08 12 41 PAGE    2
CARD  11
CARD  12
CARD  13
CARD  14
CARD  15
CARD  16
CARD  17
CARD  18
CARD  19
CARD  20
SO FAR  20 CARDS WERE PRINTED
```

شكل (٦ – ١٤)

(٤) الأسطر من ٨٨ إلى ٩٠ فى شكل (٦ – ١٣) بها حقول منقحة HEADING- TIME , HEADING- JULIAN فى مخزن العمل . يقبل التاريخ المعتاد فى WS- REGULAR- DATE لان النظام يحفظ التاريخ على هيئة yymmdd (هذه هى الصيغة الأكثر راحة إذا كانت هناك حاجة للترتيب طبقا للتاريخ ، والسنة التأثير الأكبر) . تنقل الحقول الجزئية day , month , year منفصلة إلى منطقة العنوان (انظر الأسطر من 92 إلى 94)؛ بحيث يمكن أن يعاد ترتيبها .

٦ – ١٠٣ البرنامج الموجود فى شكل (٦ – ١٥)، هو نفسه مثل البرنامج الموجود فى شكل (٦ – ١٣) ، فيما عدى أن به WITH DEBUGGING MODE محددة فى مقطع كمبيوتر المصدر ، وأن عبارات DISPLAY مضافة إلى جزء الإجراءات؛ للمساعدة فى تصحيح البرنامج. ما مخرجات عبارات DISPLAY هذه ؟

```
       PROCEDURE DIVISION.

           MOVE "NO " TO END-OF-CARDS-SWITCH
           MOVE ZERO  TO WS-PAGE-NUMBER
                         WS-CARD-COUNT

           ACCEPT WS-REGULAR-DATE        FROM DATE
           ACCEPT HEADING-JULIAN         FROM DAY
           ACCEPT HEADING-TIME           FROM TIME

      D    DISPLAY "STARTING VALUE OF END-OF-CARDS-SWITCH="
      D        END-OF-CARDS-SWITCH
      D    DISPLAY "STARTING VALUE OF WS-PAGE-NUMBER AND CARD-COUNT="
      D        WS-PAGE-NUMBER  WS-CARD-COUNT
      D    DISPLAY "VALUE OF WS-REGULAR-DATE=" WS-REGULAR-DATE

      D    DISPLAY "VALUE OF HEADING-TIME="     HEADING-TIME
      D    DISPLAY "VALUE OF JULIAN DATE="      HEADING-JULIAN


           MOVE WS-MM                    TO HEADING-MM
           MOVE WS-DD                    TO HEADING-DD
           MOVE WS-YY                    TO HEADING-YY

           OPEN    INPUT                 CARD-FILE
                   OUTPUT                PRINT-FILE

           PERFORM INPUT-A-RECORD

           PERFORM PRODUCE-PAGE-HEADING

           PERFORM PRODUCE-80-80-LISTING
               UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"

           CLOSE   CARD-FILE
                   PRINT-FILE
           STOP RUN
           .

       INPUT-A-RECORD.

           READ CARD-FILE RECORD
               INTO WS-CARD-AREA
               AT END
                   MOVE "YES" TO END-OF-CARDS-SWITCH
           .
      D    DISPLAY "JUST READ THE FOLLOWING CARD: " WS-CARD-AREA
      D    DISPLAY "END-OF-CARDS-SWITCH IS: " END-OF-CARDS-SWITCH
      D    .


       PRODUCE-80-80-LISTING.

           ADD 1 TO WS-CARD-COUNT

           WRITE PRINT-LINE
               FROM WS-LINE-AREA
               AFTER  ADVANCING 1 LINE
               AT END-OF-PAGE
                   PERFORM PRODUCE-PAGE-FOOTING
                   PERFORM PRODUCE-PAGE-HEADING
```

```
         PERFORM INPUT-A-RECORD

     PRODUCE-PAGE-HEADING.

D        DISPLAY "** ENTERING PRODUCE-PAGE-HEADING, CARD COUNT="
D            WS-CARD-COUNT

         ADD 1 TO WS-PAGE-NUMBER
         MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE-NUMBER
         WRITE PRINT-LINE
             FROM WS-HEADING-AREA
             AFTER ADVANCING PAGE


     PRODUCE-PAGE-FOOTING.

D        DISPLAY "** ENTERING PRODUCE-PAGE-FOOTING, CARD COUNT="
D            WS-CARD-COUNT

         MOVE WS-CARD-COUNT TO WS-FOOTING-COUNT

         WRITE PRINT-LINE
             FROM WS-FOOTING-AREA
             AFTER ADVANCING 2 LINES
```

شكل (٦ - ١٥)

انظر شكل (٦ - ١٦) لاحظ أن مترجم كوبل IBM OS/VS يحذف إشارة العمليات من الحقول العددية المعروضة من عبارة DISPLAY إذا كانت هذه الحقول موجبة ، وعلى هذا يطبع WS- CARD- COUNT فى صورة مقروءة . لاحظ كذلك أن اعداد COMP-3 تحولت إلى استخدام DISPLAY ؛ بغرض طباعتها بعبارة DISPLAY .

أدرس استخدام DISPLAY هذا والمخرجات المصاحبة له جيدا ؛ فهذا يمكن أن يوفر لك كثيرًا من وقت التصحيح، ولاحظ كذلك بصفة – خاصة كيف يعكس شكل (٦ - ١٦) ترتيب تنفيذ الكمبيوتر لعبارات البرنامج .

```
STARTING VALUE OF END-OF-CARDS-SWITCH=NO
STARTING VALUE OF WS-PAGE-NUMBER AND CARD-COUNT=000000
VALUE OF WS-REGULAR-DATE=830509
VALUE OF HEADING-TIME=13 14 00 72
VALUE OF JULIAN DATE=83 129
JUST READ THE FOLLOWING CARD: CARD   1
END-OF-CARDS-SWITCH IS: NO
** ENTERING PRODUCE-PAGE-HEADING, CARD COUNT=000
JUST READ THE FOLLOWING CARD: CARD   2
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD   3
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD   4
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD   5
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD   6
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD   7
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD   8
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD   9
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD  10
```

```
END-OF-CARDS-SWITCH IS: NO
** ENTERING PRODUCE-PAGE-FOOTING, CARD COUNT=010
** ENTERING PRODUCE-PAGE-HEADING, CARD COUNT=010
JUST READ THE FOLLOWING CARD: CARD  11
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD  12
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD  13
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD  14
END-OF-CARDS-SWITCH IS: NO
JUST READ THE FOLLOWING CARD: CARD  15
```

شكل (٦ - ١٦)

٦ - ١٠٤ بين كيف تنقل A والموصوف بالصورة PIC SV999 إلى B ، والذى له الصورة SV9 مع التقريب .

COMPUTE B ROUNDED = A

بالرغم من الشكل غير اللطيف فإن مايلى يؤدى نفس العمل .

ADD ZERO A GIVING B ROUNDED

٦ - ١٠٥ بين كيف يبدو الحقل المستقبل بعد كل عملية نقل تالية :

| | *Sending Field Value* | *Receiving Field PICTURE* |
|---|---|---|
| (*a*) | 123.456 | S9(5)V9(4) |
| (*b*) | 123.456 | S9(5)V9 |
| (*c*) | "STRING" | X(4) JUSTIFIED RIGHT |
| (*d*) | "STRING" | X(4) |
| (*e*) | "DOG" | X(5) |
| (*f*) | "DOG" | X(5) JUSTIFIED RIGHT |
| (*g*) | "DOG" | 9(3) |
| (*h*) | HIGH-VALUES | 9(3) |
| (*i*) | HIGH-VALUES | X(3) |
| (*j*) | SPACES | 9(3) |
| (*k*) | ALL "$" | X(5) |
| (*l*) | 123.456 | S99V99 |
| (*m*) | −123.456 | 999V999 |
| (*n*) | −123.456 | $$$$.99− |
| (*o*) | −123456 | $$$$$$.99BCR |

00123$_\wedge$4560 (a)

00123$_\wedge$4 (b)

RING (c)

STRI (d)

DOGbb (e)

(f) bb DOG

(g) غير صحيح.. فالنقل يخلط بين العددى وغير العددى .

(h) غير صحيح HIGH- VALUES غير عددى، وتنقل إلى حقل معرف بواسطة PIC X فقط .

(i) ٣ بايت لأعلى رمز فى تسلسل التتابع .

(l) غير صحيح SPACES تنقل إلى عناصر حرفية أو حرفية عددية فقط

(k) $$$$$

(l) $23_{\wedge}45$ ( يحدث حذف من ناحية اليسار، ومن ناحية اليمين ) .

(m) $123_{\wedge}456$ ( تنتج القيمة المطلقة ، حيث يفتقر الحقل المستقبل إلى وجود إشارة ) .

(n) $123.45-

(o) $23456.00bCR ( يحذف الرقم الموجود على أقصى اليسار ) .

٦ – ١٠٦ بمعرفة مايلى :

```
01  A.
    05  B     PIC X(4)     VALUE "THIS".
    05  C     PIC X(2)     VALUE "IS".
    05  D     PIC X(3)     VALUE "FUN".

01  W.
    05  X     PIC X(3).
    05  Y     PIC X(2).
    05  Z     PIC X(3).
```

اوجد محتويات: (أ) X (ب) Y (جـ) Z بعد نقل A إلى W .

يعالج الكوبل النقل لمجموعة عناصر، كما لو كان نقلا لحقل كبير من النوع الحرفى عددى . وهنا A يشغل ٩ بايت، وW يشغل ٨ بايت ، وعلى هذا يحذف البايت الموجود على أقصى اليمين من A (وهو البايت الموجود على أقصى اليمين من D) يحذف .

(*a*) "THI" (*b*) "SI" . (*c*) "SFU"

٦ – ١٠٧ انقد مايلى :

```
01  A.
    05  B     PIC S9(3)V99     COMP-3.
    05  C     PIC S99          COMP.
    05  D     PIC S99V9        DISPLAY.

MOVE ZEROS TO A
```

حيث إن A مجموعة عناصر فإنها تعامل كعنصر حرفى عددى له الاستخدام DISPLAY ، وتنقل أصفار الى حقول A. ويحدث هذا للعنصر D ( حيث إن استخدامه من نوع DISPLAY ) ، لكنه لايحدث للعنصرين C , B .

الطريقة الصحيحة هى:

```
MOVE ZEROS TO  B  C  D
```

والتى تنقل أصفارا إلى كل حقل فردى ، مع التحويل التلقائى إلى الاستخدام USAGE الصحيح .

٦ - ١٠٨ انقد مايلى :

```
. . . . . . . . . . . . . . . . . . . . . . .
STOP RUN
DISPLAY "ALL FILES ARE NOW CLOSED"
```

العبارات صحيحة من ناحية التكوين ، ولكن لايمكن لعبارة DISPLAY أن تنفذ . عندما تنفذ عبارة STOP RUN .. يتوقف الكمبيوتر عن تنفيذ التعليمات من البرنامج الحالى .

٦ - ١٠٩ افرض أن اقصى أجر فى الساعة فى إحدى الشركات هو 55.00 دولار . افرض أن أقصى عدد ساعات عمل مسموح به فى الأسبوع 60 ساعة ( مع دفع مرة ونصف من معدل الأجر المعتاد للساعات التى تزيد عن 40 ساعة ) . إذا كان عدد العاملين بالشركة 1000 عامل ، اكتب PICTURE الخاصة بـ TOTAL- WEEKLY- PAYROLL بحيث لايحدث خطأ فى الحجم .

إذا عمل أحد العاملين اقصى ساعات عمل، باقصى معدل أجر ، فإن أجر العامل الأسبوعى سيكون

$$(40 \times 55.00) + (20 \times 55.00 \times 1.5) = 3850.00 \text{ dollars}$$

وعلى هذا .. لايمكن أن يتعدى إجمالى الأجر الأسبوعى لجميع العاملين :

$$1000 \times 3850.00 = 3{,}850{,}000.00 \text{ dollars}$$

وعلى هذا .. فإن PIC S9(7)V99 تكون كافية .

٦ - ١١٠ عبر عن العلاقات الجبرية التالية باستخدام عبارة COMPUTE :

$$(a)\quad a = \frac{b}{c} + b - (c \times d) \qquad (b)\quad a = \frac{b+c}{d} - \frac{c}{e+f} \qquad (c)\quad d = \frac{(a+b)^2}{a-b} + a - (b+c)$$

```
(a)    COMPUTE A = B / C + B - C * D
(b)    COMPUTE A = (B + C) / D - C / (E + F)
(c)    COMPUTE D = (A + B) ** 2 / (A - B) + A - (B + C)
```

```
       01  MAILING-INPUT.
           05  MAILING-NAME             PIC X(20).
           05  MAILING-ADDRESS          PIC X(20).
           05  MAILING-CITY             PIC X(20).
           05  MAILING-STATE            PIC X(2).
           05  MAILING-ZIP              PIC X(5).
           05  FILLER                   PIC X(13).

       FD  MAILING-LABEL-FILE
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .
       01  LABEL-LINE                   PIC X(132).

       WORKING-STORAGE SECTION.

       01  PROGRAM-SWITCHES-AND-COUNTERS.
           05  END-OF-FILE-SW           PIC X(3).
           05  WS-LABEL-COUNT           PIC S9(3)     COMP-3.

       01  WS-NAME-LINE.
           05  NAME-1                   PIC X(30)     VALUE ALL "*".
           05  FILLER                   PIC X(10)     VALUE SPACES.
           05  NAME-2                   PIC X(30)     VALUE ALL "*".
           05  FILLER                   PIC X(62)     VALUE SPACES.

       01  WS-ADDRESS-LINE.
           05  ADDRESS-1                PIC X(30)     VALUE ALL "*".
           05  FILLER                   PIC X(10)     VALUE SPACES.
           05  ADDRESS-2                PIC X(30)     VALUE ALL "*".
           05  FILLER                   PIC X(62)     VALUE SPACES.

       01  WS-CITY-STATE-LINE.
           05  CITY-1                   PIC X(20)     VALUE ALL "*".
           05  FILLER                   PIC X         VALUE SPACES.
           05  STATE-1                  PIC XX        VALUE ALL "*".
           05  FILLER                   PIC X(2)      VALUE SPACES.
           05  ZIP-1                    PIC X(5)      VALUE ALL "*".
           05  FILLER                   PIC X(10)     VALUE SPACES.
           05  CITY-2                   PIC X(20)     VALUE ALL "*".
           05  FILLER                   PIC X         VALUE SPACES.
           05  STATE-2                  PIC XX        VALUE ALL "*".
           05  FILLER                   PIC X(2)      VALUE SPACES.
           05  ZIP-2                    PIC X(5)      VALUE ALL "*".
           05  FILLER                   PIC X(62)     VALUE SPACES.

       01  WS-COUNT-LINE.
           05  PRINTED-COUNT            PIC Z,ZZ9.
           05  FILLER                   PIC X(15)
                                        VALUE " LABELS PRINTED".
           05  FILLER                   PIC X(112)    VALUE SPACES.

       PROCEDURE DIVISION.

           MOVE "NO " TO END-OF-FILE-SW
           MOVE ZERO  TO WS-LABEL-COUNT

           OPEN    INPUT                MAILING-MASTER
                   OUTPUT               MAILING-LABEL-FILE
```

شكل (٦ - ١٧) تكملة

```
           PERFORM 030-PRINT-2-LABELS
               2 TIMES

           PERFORM 020-PRODUCE-MAILING-LABELS
               UNTIL END-OF-FILE-SW IS EQUAL TO "YES"

           PERFORM 040-PRODUCE-COUNT-LINE

           CLOSE   MAILING-MASTER
                   MAILING-LABEL-FILE
           STOP RUN
           .

       010-INPUT-2-RECORDS.

           MOVE SPACES TO WS-NAME-LINE
                          WS-ADDRESS-LINE
                          WS-CITY-STATE-LINE

           READ MAILING-MASTER RECORD
               AT END
                   MOVE "YES" TO END-OF-FILE-SW
           .
           IF END-OF-FILE-SW = "NO"

               ADD 1 TO WS-LABEL-COUNT
               MOVE MAILING-NAME        TO NAME-1
               MOVE MAILING-ADDRESS     TO ADDRESS-1
               MOVE MAILING-CITY        TO CITY-1
               MOVE MAILING-STATE       TO STATE-1
               MOVE MAILING-ZIP         TO ZIP-1

               READ MAILING-MASTER RECORD
                   AT END
                       MOVE "YES" TO END-OF-FILE-SW
           .
           IF END-OF-FILE-SW EQUAL "NO"

               ADD 1 TO WS-LABEL-COUNT
               MOVE MAILING-NAME        TO NAME-2
               MOVE MAILING-ADDRESS     TO ADDRESS-2
               MOVE MAILING-CITY        TO CITY-2
               MOVE MAILING-STATE       TO STATE-2
               MOVE MAILING-ZIP         TO ZIP-2
           .

       020-PRODUCE-MAILING-LABELS.

           PERFORM 010-INPUT-2-RECORDS

           PERFORM 030-PRINT-2-LABELS
           .

       030-PRINT-2-LABELS.

           WRITE LABEL-LINE
               FROM WS-NAME-LINE
               BEFORE ADVANCING 1 LINE
           WRITE LABEL-LINE
```

شكل (٦ - ١٧) تكملة

```
                FROM WS-ADDRESS-LINE
                BEFORE ADVANCING 1 LINE
            WRITE LABEL-LINE
                FROM WS-CITY-STATE-LINE
                BEFORE ADVANCING 2 LINES
            .

        040-PRODUCE-COUNT-LINE.

            MOVE WS-LABEL-COUNT TO PRINTED-COUNT
            WRITE LABEL-LINE
                FROM WS-COUNT-LINE
                AFTER ADVANCING 2 LINES
```

شكل (٦ – ١٧) تكملة

```
NATHAN CONVERSE                          MIKE PERELMAN
709 N. SIMPSON ST.                       213 E. SOUTH ST.
MIAMI                 FL   10022         HANOVER               MD   23451

BARNEY PERELMAN                          KATHY BUCHER
782 N. SOUTH ST.                         666 W. EAST ST.
NEW YORK              NY   10012         PHILADELPHIA          PA   34256

ABE SHARP
783 S. NORTH ST.
BALTIMORE             MD   22145

     5 LABELS PRINTED
```

شكل (٦ – ١٨)

٦ – ١١٣ اكتب برنامجا يحسب إحصائيات مبيعات مبنية على عينة من المبيعات الشهرية لأحد المحلات متعدد المخازن . تحتوى المدخلات على سجلات منطقية تشمل مايلى : معرفاً للمخزن، ورمزاً للموقع، وكمية المبيعات الشهرية . وتحتوى المخرجات على تقرير مطبوع يبين سطرا تفصيليا، يشمل معلومات عن كل مخزن ، يتبعه سطر نهائى يعطى ، العينة ، متوسط المبيعات والانحراف المعيارى للمبيعات .

تحسب كميات العينة من العلاقة التالية :

$$\text{sample total} = s_1 + s_2 + \cdots + s_n$$

$$\text{average} = \frac{\text{sample total}}{n} = \frac{s_1 + s_2 + \cdots + s_n}{n}$$

$$\text{standard deviation} = \sqrt{\frac{n(s_1^2 + s_2^2 + \cdots + s_n^2) - (s_1 + s_2 + \cdots + s_n)^2}{n(n-1)}}$$

حيث n عدد المخازن المأخوذة فى العينة، و $S_i$ كمية مبيعات المخزن رقم i .

**الإجابة** : انظر شكل (٦ – ١٩) ( توجد مخرجات تقليدية فى شكل ٦ – ٢٠ ) . بالرغم من أنه غير مطلوب ، إلا أن هذا البرنامج يطبع سطر نهاية يعطى إجماليًا جاريًا للمبيعات .

```
       IDENTIFICATION DIVISION.

       PROGRAM-ID. SALESTAT.
       AUTHOR.  LARRY NEWCOMER.
       INSTALLATION.  PENN STATE UNIVERSITY--YORK CAMPUS.
       DATE-WRITTEN.  MAY 1983.
       DATE-COMPILED. MAY  9,1983.
       SECURITY.  NONE.

      *    SALESTAT PRODUCES A LISTING OF MONTHLY SALES BY RETAIL
      *    STORE.  IT PRINTS A FINAL LINE SHOWING THE TOTAL SALES,
      *    THE AVERAGE SALES, AND THE STANDARD DEVIATION FOR SALES.

      *    SALESTAT PROVIDES A GOOD ILLUSTRATION OF THE USE OF
      *    THE COMPUTE STATEMENT.

       ENVIRONMENT DIVISION.

       CONFIGURATION SECTION.
       SOURCE-COMPUTER. IBM-370.
       OBJECT-COMPUTER. IBM-370.
       INPUT-OUTPUT SECTION.
       FILE-CONTROL.
           SELECT SALES-FILE             ASSIGN TO SALES.
           SELECT SALES-REPORT           ASSIGN TO SALELIST.

       DATA DIVISION.

       FILE SECTION.

       FD  SALES-FILE
           RECORD CONTAINS 80 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .
       01  SALES-INPUT                   PIC X(80).

       FD  SALES-REPORT
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED
           LINAGE IS 10
               WITH FOOTING AT 8
               LINES AT TOP 2
               LINES AT BOTTOM 2


       01  SALES-LINE                   PIC X(132).

       WORKING-STORAGE SECTION.

       01  WS-DATE-AND-TIME-AREAS.
           05  WS-REGULAR-DATE.
               10  WS-YY                PIC 99.
               10  WS-MM                PIC 99.
```

شكل (٦ - ١٩)

```
                10  WS-DD                                       PIC 99.

        01  PROGRAM-SWITCHES-COUNTERS-SUMS.
            05  END-OF-SALES-SWITCH      PIC X(3).
            05  WS-PAGE-NUMBER           PIC S9(3)        COMP-3.
            05  WS-TOTAL-SALES           PIC S9(7)V99     COMP-3.
            05  WS-NUMBER-STORES         PIC S9(5)        COMP-3.
            05  WS-SUM-SALES-SQUARED     PIC S9(15)V99    COMP-3.

        01  WS-SALES-AREA.
            05  WS-SALES-STORE-ID        PIC X(5).
            05  WS-SALES-LOCATION-CODE   PIC XX.
            05  WS-SALES-SALES           PIC S9(6)V99.
            05  FILLER                   PIC X(65).

        01  WS-LINE-AREA.
            05  WS-LINE-STORE-ID         PIC X(5).
            05  FILLER                   PIC X(5)         VALUE SPACES.
            05  WS-LINE-LOCATION-CODE    PIC XX.
            05  FILLER                   PIC X(5)         VALUE SPACES.
            05  WS-LINE-SALES            PIC ZZZ,ZZZ.99.
            05  FILLER                   PIC X(105)       VALUE SPACES.

        01  WS-HEADING-AREA-1.
            05  FILLER                   PIC X(1)         VALUE SPACES.
            05  FILLER                   PIC X(20)
                                         VALUE "SALES STATISTICS".
            05  HEADING-MM               PIC Z9.
            05  FILLER                   PIC X            VALUE "/".
            05  HEADING-DD               PIC 99.
            05  FILLER                   PIC X            VALUE "/".
            05  HEADING-YY               PIC 99.
            05  FILLER                   PIC X(4)         VALUE SPACES.
            05  FILLER                   PIC X(6)         VALUE "PAGE ".
            05  WS-HEADING-PAGE-NUMBER   PIC ZZ9.
            05  FILLER                   PIC X(90)        VALUE SPACES.

        01  WS-HEADING-AREA-2.
            05  FILLER                   PIC X(8)         VALUE "STORE ID".
            05  FILLER                   PIC X(2)         VALUE SPACES.
            05  FILLER                   PIC X(12)        VALUE "LOC".
            05  FILLER                   PIC X(10)        VALUE "SALES".
            05  FILLER                   PIC X(100)       VALUE SPACES.

        01  WS-FOOTING-AREA.
            05  FILLER                   PIC X(15)        VALUE SPACES.
            05  WS-FOOTING-TOTAL         PIC Z,ZZZ,ZZZ.99.
            05  FILLER                   PIC X(105)       VALUE " *".

        01  WS-FINAL-LINE-1.
            05  FILLER                   PIC X(15)        VALUE SPACES.
            05  WS-FINAL-TOTAL           PIC Z,ZZZ,ZZZ.99.
            05  FILLER                   PIC X(105)       VALUE " **".

        01  WS-FINAL-LINE-2.
            05  FILLER                   PIC X(10)        VALUE "AVERAGE--".
            05  WS-AVERAGE-SALES         PIC ZZZ,ZZZ.99.
            05  FILLER                   PIC X(6)         VALUE " SD-- ".
            05  WS-SD                    PIC ZZZ,ZZZ.99.
            05  FILLER                   PIC X(96)        VALUE SPACES.

        PROCEDURE DIVISION.

            MOVE "NO " TO END-OF-SALES-SWITCH
            MOVE ZERO  TO WS-PAGE-NUMBER
                          WS-TOTAL-SALES
```

```
                             WS-NUMBER-STORES
                             WS-SUM-SALES-SQUARED

           ACCEPT WS-REGULAR-DATE        FROM DATE

           MOVE WS-MM                    TO HEADING-MM
           MOVE WS-DD                    TO HEADING-DD
           MOVE WS-YY                    TO HEADING-YY

           OPEN    INPUT                 SALES-FILE
                   OUTPUT                SALES-REPORT

           PERFORM GET-SALES-RECORD

           PERFORM PRODUCE-PAGE-HEADING

           PERFORM PRODUCE-SALES-LINE
               UNTIL END-OF-SALES-SWITCH IS EQUAL TO "YES"

           PERFORM PRODUCE-FINAL-LINES

           CLOSE   SALES-FILE
                   SALES-REPORT
           STOP RUN
           .

       GET-SALES-RECORD.

           READ SALES-FILE RECORD
               INTO WS-SALES-AREA
               AT END
                   MOVE "YES" TO END-OF-SALES-SWITCH
           .


       PRODUCE-SALES-LINE.

           ADD 1 TO WS-NUMBER-STORES
           COMPUTE WS-SUM-SALES-SQUARED =   WS-SUM-SALES-SQUARED
                                          + WS-SALES-SALES ** 2

           ADD WS-SALES-SALES TO WS-TOTAL-SALES

           MOVE WS-SALES-STORE-ID        TO WS-LINE-STORE-ID
           MOVE WS-SALES-LOCATION-CODE   TO WS-LINE-LOCATION-CODE
           MOVE WS-SALES-SALES           TO WS-LINE-SALES

           WRITE SALES-LINE
               FROM WS-LINE-AREA
               AFTER  ADVANCING 2 LINES
               AT END-OF-PAGE
                   PERFORM PRODUCE-PAGE-FOOTING
                   PERFORM PRODUCE-PAGE-HEADING
           .

           PERFORM GET-SALES-RECORD
           .

       PRODUCE-PAGE-HEADING.

```

شكل (٦ - ١٩) تكملة

```
           ADD 1 TO WS-PAGE-NUMBER
           MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE-NUMBER
           WRITE SALES-LINE
               FROM WS-HEADING-AREA-1
               AFTER ADVANCING PAGE
           WRITE SALES-LINE

           FROM WS-HEADING-AREA-2
           AFTER ADVANCING 1 LINE


   PRODUCE-PAGE-FOOTING.

       MOVE WS-TOTAL-SALES TO WS-FOOTING-TOTAL

       WRITE SALES-LINE
           FROM WS-FOOTING-AREA
           AFTER ADVANCING 2 LINES


   PRODUCE-FINAL-LINES.

       MOVE WS-TOTAL-SALES          TO WS-FINAL-TOTAL
       WRITE SALES-LINE
           FROM WS-FINAL-LINE-1
           AFTER ADVANCING 2 LINES

       COMPUTE WS-AVERAGE-SALES = WS-TOTAL-SALES / WS-NUMBER-STORES

       COMPUTE WS-SD =((WS-NUMBER-STORES * WS-SUM-SALES-SQUARED
                                         - WS-TOTAL-SALES ** 2)
                               /
                (WS-NUMBER-STORES * (WS-NUMBER-STORES - 1))) ** .5

       WRITE SALES-LINE
           FROM WS-FINAL-LINE-2
           AFTER ADVANCING 2 LINES

```

شكل (٦ - ١٩) تكملة

```
 SALES STATISTICS      5/09/83      PAGE      1
STORE ID  LOC            SALES
00001     AA          1,000.00
00002     BB          2,000.00
00003     CC          3,000.00
                      6,000.00 *

 SALES STATISTICS      5/09/83      PAGE      2
STORE ID  LOC            SALES
00004     DD          4,000.00
00005     EE          5,000.00
00006     FF          6,000.00
                     21,000.00 *

 SALES STATISTICS      5/09/83      PAGE      3
STORE ID  LOC            SALES
00007     GG          7,000.00
00008     HH          8,000.00
                     36,000.00 **
AVERAGE--    4,500.00 SD--    2,449.49
```

شكل (٦ - ٢٠)

## تمارين برمجة

٦ - ١١٤ بمعرفة الأساس P المستثمر فى حساب إدخار بمعدل فائدة I يحسب بفائدة مركبة عدد C من المرات خلال السنة ، فإن إجمالى المبلغ المتراكم T خلال N سنة هو :

$$T = P * \left(1 + \frac{I}{C}\right)^{C*N}$$

اكتب برنامجا بلغة الكوبل يقوم بإدخال C, I, P، ويطبع تقريرا مبينا T, N, C, I, P . اطبع عدداً من الأسطر يناظر عدد سطور المدخلات . صمم سجلات المدخلات وتخطيط التقرير ، واستخدم جزء LINAGE ؛ للتأكد من أن ( ١ ) وجود ٢٠ سطر تفصيليًا مع مسافة مزدوجة بينها فى كل صفحة منطقية، (٢) وجود هامش علوى وهامش سفلى بعرض ثلاثة سطور لكل منها (٣) عنوان وعناوين أعمدة فى بداية جسم كل صفحة (بما فى ذلك رقم الصفحة وتاريخ ووقت التنفيذ).

٦ - ١١٥ اكتب برنامجا بلغة الكوبل، يدخل سجلات مخزون، طول السجل ٨٠ بايت، وله الشكل التالى :

الأعمدة ١ - ٦ : رقم العنصر

الاعمدة من ٧ - ٢٦ : وصف العنصر

الاعمدة من ٢٧ – ٣١ : الكمية الموجودة

الاعمدة من ٣٢ – ٣٨ : التكلفة ( مع ترك خانتين للكسر العشرى)

البقية : غير مستخدم

اطبع تقريرا به سطر لكل عنصر، مبينا مايلى :

| item # | description | quantity on hand | cost | worth |
|---|---|---|---|---|

حيث worth هى حاصل ضرب الكمية الموجودة quantity on hand فى التكلفة cost . أطبع فى نهاية التقرير سطرا نهائيا يبين: (١) إجمالى قيمة worth المخزون (أى مجموع القيمة لكل العناصر ) (٢) ومتوسط تكلفة العنصر (خارج قسمة إجمالى المخزون على مجموع الكمية الموجودة ) .

صمم الصفحة المنطقية بنفسك على أن تشمل : (١) عناوين، (٢) منطقة نهاية تظهر فيها القيمة الحالية لإجمالى الكمية الموجودة (انظر شكل (٦ – ٢٠)، (٣) رقمًا للصفحة وتاريخ ووقت التنفيذ فى بداية كل صفحة .

٦ – ١١٦ لدى إحدى المكاتب الاستشارية معدل داخلى internal rate لكل العاملين فيها (مثل هذا المعدل مايدفعه المكتب لكل عامل فى الساعة)، ولديها معدل خارجى external rate (مايدفعه العميل لكل ساعة عمل للعامل بالمكتب) . بمعرفة المدخلات التالية :

الأعمدة من ١ – ٤ : رقم تعريف العامل

الأعمدة من ٥ – ٩ : المعدل الداخلى ( خانتين للكسر العشرى )

الأعمدة من ١٠ – ١٤ : المعدل الخارجى (خانتين للكسر)

الأعمدة من ١٥ – ١٧ : عدد ساعات العمل الاسبوعية ( خانة للكسر العشرى )

اطبع تقريرا للرواتب يبين :

| employee ID | hours worked | internal rate | payroll amount | external rate | amount charged | gross profit |
|---|---|---|---|---|---|---|

حيث

payroll amount = hours * internal rate
amount charged = hours * external rate
gross profit = amount charged − payroll amount

وفى نهاية التقرير ، اطبع سطرا يبين إجمالى قيمة الرواتب ، وإجمالى القيمة المطلوبة من العملاء ، ومتوسط إجمالى الربح لكل عامل بالمكتب . يجب أن تحتوى الصفحة المنطقية على مايلى :

(١) عناوين (٢) نهاية تبين إجمالى عدد العاملين الذى أجرى لهم تشغيل حتى الآن (٣) رقم الصفحة وتاريخ ووقت تنفيذ البرنامج فى بداية كل صفحة .

الفصل السابع

# منطق البرنامج

# Program Logic

## ٧ - ١ التصميم المنطقى : خرائط المسار المرتبة

يشير منطق البرنامج program logic الى ترتيب تنفيذ عبارات البرنامج ( على عكس ترتيبها الواقعى أو الطبيعى الموجوده فيه فى قائمة المصدر ) . خريطة المسار المرتبة structured flowchart هى رسم يحدد ترتيب العمليات فى الخوارزمى . وعلى هذا .. يمكن استخدام خريطة مسار مرتبة فى تصميم منطق جزء حل المشكلة من البرنامج ، قبل أن يكتب البرنامج فعلا بالكوبل ( أو بأى لغة أخرى ) . إذا ما حفظت خريطة المسار مجددة ، تعكس التغييرات فى البرنامج ، فإنها تخدم كذلك كتوثيق مفيد للبرنامج .

والرموز النمطية لخريطة المسار موضحة فى شكل ٧ . ١ .

## ٧ - ٢ هياكل منطق البرنامج

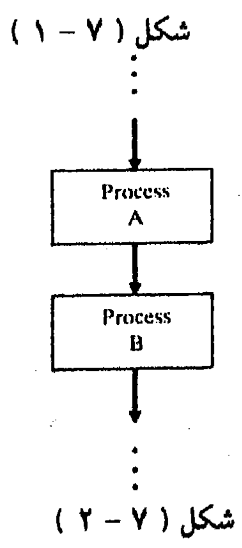
يمكن رؤية أن منطق أى خوارزمى أو برنامج كمبيوتر يمكن تحقيقه كخليط من ثلاثة هياكل منطقية logic structures أساسية .

## هيكل التتابع ( مكون التتابع )

هذا هو تنفيذ تتابعى لعمليات أو تعليمات ، واحدة تلو الأخرى . وفى برنامج الكمبيوتر المحتوى على هيكل تتابع .. يتفق الترتيب المنطقى للعبارات مع ترتيبها الطبيعى . ويبين شكل ٧ . ٢ خريطة مسار لهيكل التتابع .

| | |
|---|---|
| | رمز النهاية أو البداية ، يحمل كلمة START ( أو STOP أو END ) داخله ، يعرف بداية (أو نهاية ) الخوارزمى . يمكن كتابة اسم الخوارزمى داخله كذلك . |
| | رمز العملية ، يحدد تشغيل الكمبيوتر لمعلومات . توصف طبيعة التشغيل داخل الرمز . |
| | رمز القرار ، يحدد قرار الكمبيوتر للاختيار بين مسارين منطقيين فى الخوارزمى . ويتميز بسهمين تاركين الرمز ، مبينا البديلين من المسارين المنطقيين . |
| | رمز العملية سبق تعريفها ، ويمثل تشغيل كمبيوتر معقد بدرجة كافية ليتطلب خريطة مسار منفصلة تصفه . يحتوى الرمز على اسم المقطع المعد له خريطة مسار منفصلة . |
| | رمز الواصل ، يمثل توصيلا لجزءين أو أكثر من خريطة المسار. |
| | رمـز مدخلات أو مخرجـات ، يحدد عملية مدخلات وعملية مخرجات . |
| | أسهم تستخدم فى توصيل الرموز السابقة فى الترتيب التى تنفذ به . |

شكل ( ٧ - ١ )




شكل ( ٧ - ٢ )

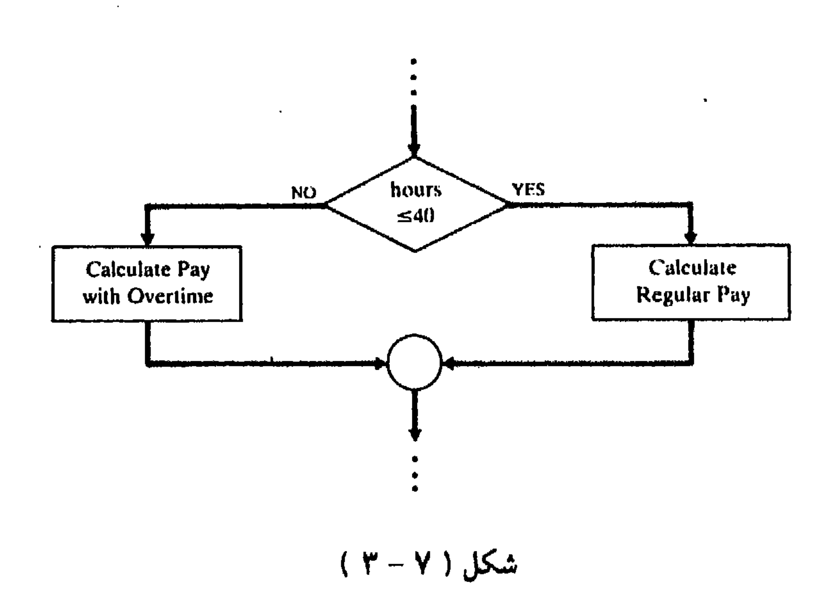
## هيكل الاختيار ( مكون الاختيار )

يختار الهيكل أحد فرعين اثنين بالضبط ( لعمليات أو مخرجات من التعليمات ) . ويعتمد اختيار الفرع على القيمة الحالية لشرط ما والتى تكون صحيح true أو خطأ false دائما .

**مثال ٧ - ١ :**

فى برنامج الرواتب .. تكون هناك حاجة لإحدى مجموعات التعليمات لحساب إجمالى راتب العامل الذى يعمل 40 ساعة أو أقل ، ومجموعة أخرى من التعليمات للعامل الذى يعمل وقتا إضافيا . وهيكل الاختيار مبين فى شكل ٧ . ٣ .

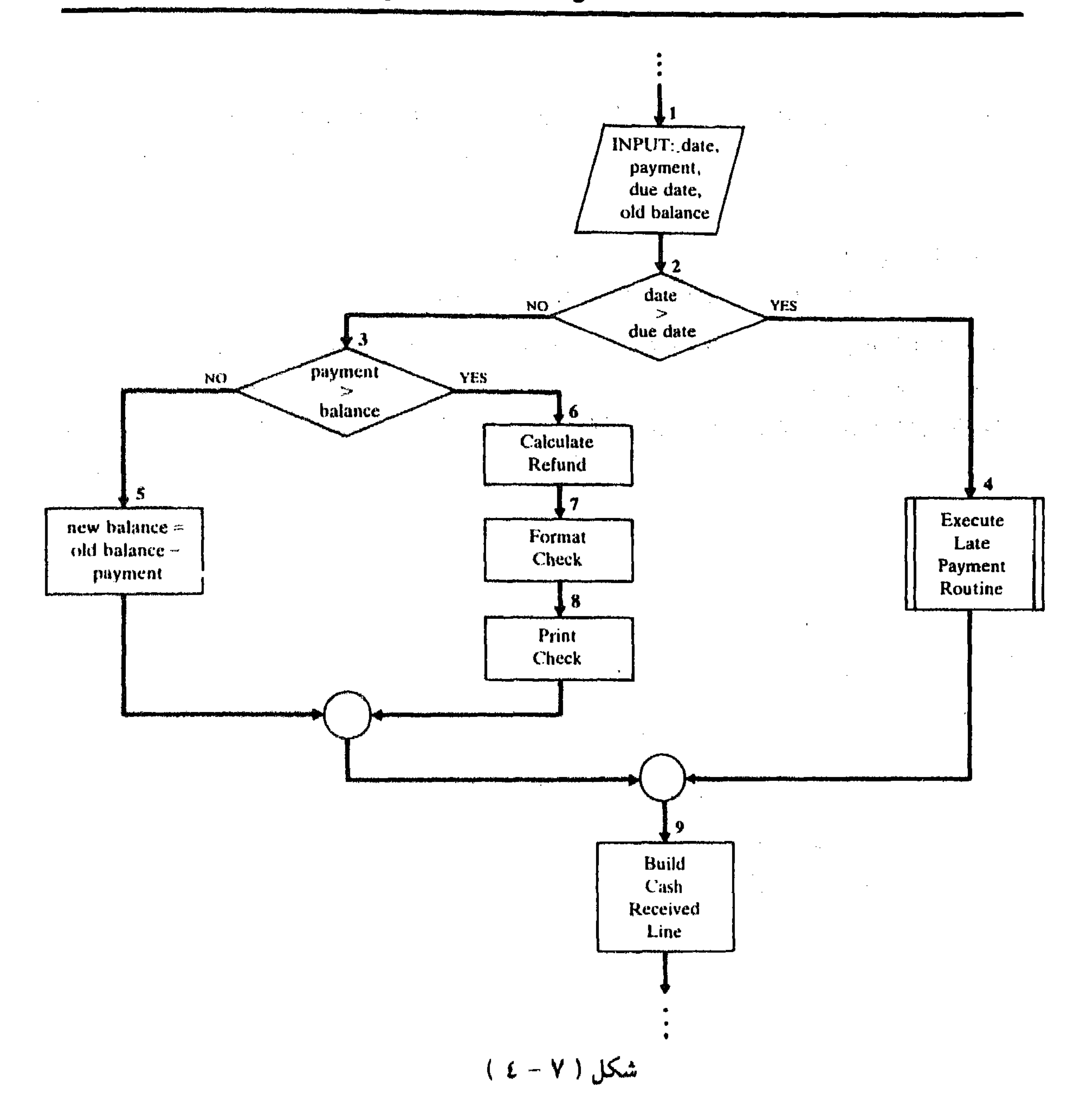
يجب أن يحتوى رمز القرار على شرط ، إما أن تكون نتيجته صحيح true أو خطأ false . وكما فى الحالة الحالية .. فإنه عادة ما يشمل الشرط عنصرى بيانات اثنين ( أو عنصر بيانات وثابت ) لتحديد ما إذا تحققت علاقة معينة ( أقل من أو يساوى هنا ) أم لا . فاذا كان الشرط صحيحا ، فيتبع السهم الخاص بكلمة YES أو TRUE فى تنفيذ هيكل الاختيار ، وإلا فيتبع السهم الخاص بكلمة NO أو FALSE .




شكل ( ٧ - ٣ )

**مثال ٧ - ٢ :**

تدمج خريطة المسار المرتبة الموجودة فى شكل ٧ . ٤ هيكل التتابع مع هيكل الاختيــار . الرمز 1 أو الرمز 2 والرمز 9 تمثل تتابــع ، الجــزء الثانى ، والذى منه الرمز 2 ، هو بنفسه هيكل اختيار ( بمسارين منطقيين بديلين خلال الرمزين 3 و 4 الرمز 4 هو عملية سابقة التعريف ) ، تمثل مجموعة ( من الممكن أن تكون معقدة ) من الإجراءات المعرفة فى خريطة مسار أخرى . الرمز 3 هو رأس هيكل اختيار آخر ، بمسارين منطقيين بديلين خلال الرمزين 5 من ناحية ، 6 و 7 و 8 من ناحية أخرى . يلاحظ أن الرموز 6 و 7 و8 تكون هيكل تتابع بنفسها ) .




شكل ( ٧ - ٤ )

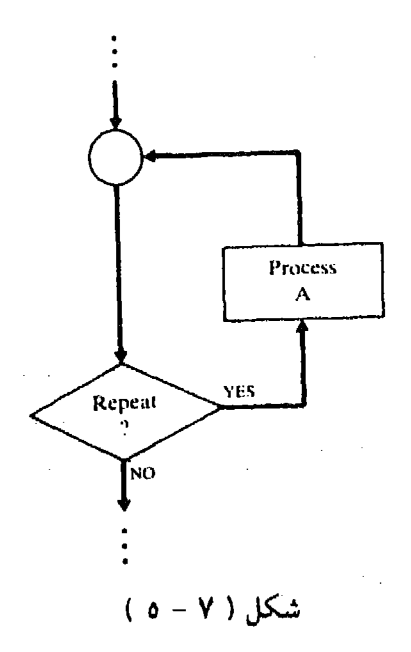
## هيكل التكرار ( مكون التكرار )

يستخدم هيكل التكرار عندما يجب تكرار عملية أو مجموعة من العمليات لعدد معين من المرات ، وهو هيكل دورة البرنامج program loop .

**مثال ٧ - ٣ :**

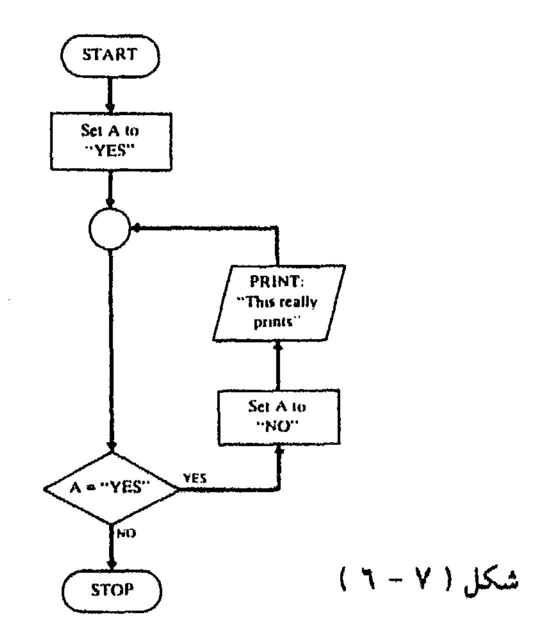
صيغة هيكل التكرار مرسوم لها خريطة مسار فى شكل ٧ . ٥ ، وعادة ما تسمى هذه الصيغة بهيكل DO WHILE . عندما يدخل الهيكل التنفيذ .. فإن أول شيء ما يحدث هو اتخاذ قرار ماذا كان سيحدث تكرار أما لا .

إذا كان القرار لا ( شرط التكرار غير صحيح ) ، فيتم الخروج من الهيكل دون تنفيذ العملية A . إذا كان القرار نعم (شرط التكرار صحيح ) تنفذ العملية A ، ويؤخذ قرار التكرار مرة أخرى .




شكل ( ٧ - ٥ )

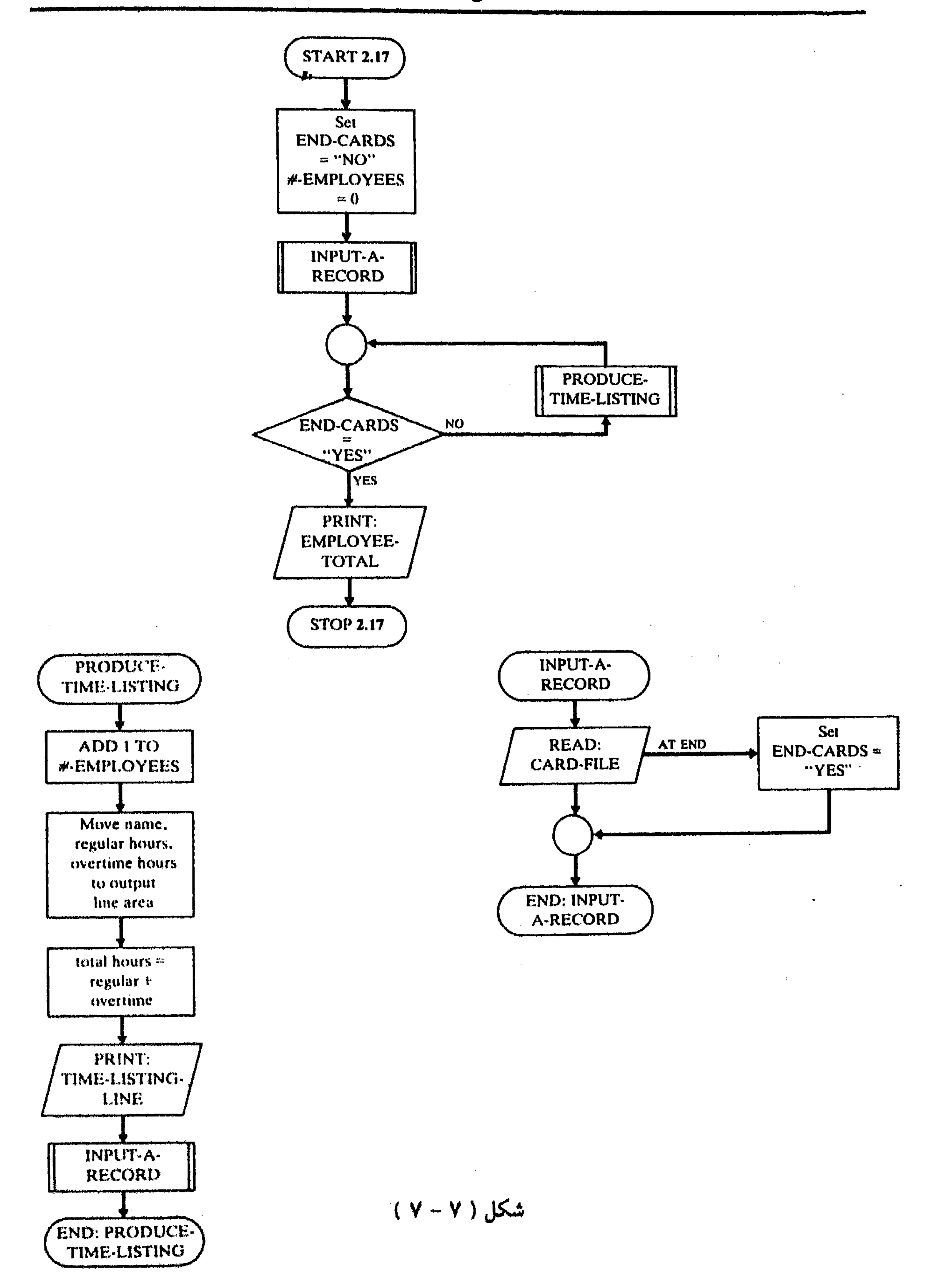
ويبين شكل ٧ - ٦ هيكلاً خاصاً DO WHILE ، يتسبب فى وضع A مساوية NO ، وطباعة PRINT عبارة This really Prints وذلك مرة واحدة . الحقيقة إن A توضع فيها NO أثناء أول تنفيذ ( وهو التنفيذ الوحيد ) لا تتسبب فى فصل هذا التنفيذ . وعندما يعاد الدخول فى المجموعة فقط .. يتسبب الشرط" A = " NO فى تجنب عمل أى تكرار أخر .




شكل ( ٧ - ٦ )

**مثال ٧ - ٤ :**

خريطة المسار المرتبة للبرنامج مثال رقم ١٧ من الفصل الثانى معطاة فى شكل ٧ . ٧ ، وتشمل هياكل تتابع وتكرار فقط .


START 2.17
Set
END-CARDS
= "NO"
#-EMPLOYEES
= 0
INPUT-A-
RECORD
PRODUCE-
TIME-LISTING
END-CARDS
=
"YES"
NO
YES
PRINT:
EMPLOYEE-
TOTAL
STOP 2.17
PRODUCE-
TIME-LISTING
ADD 1 TO
#-EMPLOYEES
Move name,
regular hours,
overtime hours
to output
line area
total hours =
regular +
overtime
PRINT:
TIME-LISTING-
LINE
INPUT-A-
RECORD
END: PRODUCE-
TIME-LISTING
INPUT-A-
RECORD
READ:
CARD-FILE
AT END
Set
END-CARDS =
"YES"
END: INPUT-
A-RECORD

شكل ( ٧ - ٧ )

لاحظ كيف توصف كل عملية سابقة التعريف فى خريطة المسار الخاصة بها ، هذه العمليات تناظر مقاطع PERFORM فى برنامج الكوبل . لاحظ كذلك أن مقطعاً معيناً من جزء الإجراءات يعمل ، مثل فتح وإغلاق الملفات ، ليس مرسوماً فى خريطة المسار .

## ٧ - ٣ هيكل التتابع فى الكوبل

هيكل التتابع هو الأسهل فى البرمجة ( بأى لغة ) . يتحقق التتابع فى الكوبل بكتابة عبارة بعد الأخرى فى جزء الإجراءات ببساطة .

## ٧ - ٤ هيكل الاختيار فى الكوبل

ينفذ هيكل الاختيار فى الكوبل بعبارة إذا IF ، والتى لها الشكل العام التالى :

```
IF condition {statement-1    } [ELSE statement-2        ]
             {NEXT SENTENCE  } [ELSE NEXT SENTENCE      ]
```

هنا يكون الشرط Condition تعبيرا أو جزءا من عبارة معقداً قليلا أو كثيرا ، ونتناوله بالتفصيل فى الأقسام من ٥ إلى ٩ فى هذا الفصل.

**مثال ٧ - ٥**

```
IF  HOURS-WORKED GREATER THAN 40.0
    COMPUTE GROSS-PAY = 40.0 * HOURLY-RATE +
                        (HOURS-WORKED - 40.0) * 1.5 * HOURLY-RATE
ELSE
    COMPUTE GROSS-PAY = HOURS-WORKED * HOURLY-RATE
```

الشرط هو : HOURS - WORKED GREATER THAN 40.0 .

ويكون صحيحا أو خطأ طبقا للمحتويات الحالية فى العنصر HOURS - WORKED . فإذا كان صحيحا ، تنفذ statement - 1 وهى ...COMPUTE ... 40.0 ، ثم يتجه الكمبيوتر بعد ذلك لتنفيذ العبارة التى تلى النقطة التى تنهى عبارة IF . اذا كان الشرط خطأ ، تهمل Statement - 1 ، وتنفذ Statement - 2 وهى ... COMPUTE ... HOURS - WORKED، ثم يتجه الكمبيوتر بعد ذلك إلى العبارة التى تلى النقطة التى تنهى عبارة IF .

بالرغم من عدم ضرورة ترجمة الترحيل الموضح ، إلا انه هو طريقة نمطية ضرورية لكتابة الشفرة . كتبت IF و ELSE فى نفس العمود ، وتم ترحيل جزء IF وجزء ELSE على التوالى تحت IF و ELSE . يجب استخدام نقطة مرتبة دائما فى تحديد نهاية كل عبارة من عبارات IF .

جزء ELSE فى عبارة IF اختيارى . فإذا حذف .. يتسبب الشرط الصحيح فى تنفيذ Statement - 1 قبل أن ينتقل الكمبيوتر إلى العبارة التى تلى النقطة التى تنهى عبارةIF ، بينما يتسبب الشرط الخطأ فى ترك الكمبيوتر Statement - 1 والانتقال مباشرة الى العبارة التالية للنقطة التى تنهى عبارة IF .

**مثال ٧ – ٦** : ليس ضروريا على الإطلاق تحديد ELSE NEXT SENTENCE ، حيث إن :

```
IF...
    PERFORM...
ELSE
    NEXT SENTENCE

WRITE...
```

يتطابق مع

```
IF...
    PERFORM...

WRITE...
```

مع أى من التشكيلين ، تنفذ PERFORM ثم WRITE إذا كان الشرط صحيحا ، بينما تنفذ WRITE فقط إذا كان الشرط خطأ .

إلا أن ELSE NEXT SENTENCE يمكن أن تكون مريحة فى بعض الأحيان ، انظر مثال ٧ – ٢٥.

**مثال ٧ – ٧**

```
IF AMOUNT-SOLD IS GREATER THAN SALES-QUOTA
    NEXT SENTENCE
ELSE
    MOVE "YES" TO SEND-MOTIVATING-LETTER-SWITCH
    SUBTRACT AMOUNT-SOLD FROM SALES-QUOTA GIVING AMOUNT-UNDER-QUOTA

WRITE REPORT-LINE
    FROM WS-SALES-PERFORMANCE-LINE
```

إذا كانت قيمة AMOUNT - SOLD أكبر من SALES - QUOTA ، تنفذ NEXT SENTENCE ، متسببة فى انتقال الكمبيوتر إلى العبارة التى تلى النقطة التى تنهى عبارة IF ، وهى ...WRITE PERIOD - LINE . إذا لم تكن AMOUNT - SOLD أكبر من SALES - QUOTA ، فيهمل الكمبيوتر جزء IF وينتقل الى جزء ELSE منفذا MOVE وSUBTRACT قبل الانتقال إلى عبارة WRITE .

لاحظ ما يتبع حذف النقطة من نهاية عبارة IF : تصبح عبارة WRITE جزاء من ELSE ، ويطبع سطر تقرير لكل بائع من البائعين الذين لم يحققوا حصص بيعهم فقط ، بدلا من طباعة سطر تقدير لكل بائع من البائعين.

**مثال٧ – ٨ :**

ليس ضروريا – على الاطلاق أن يحدد NEXT SENTENCE فى جزء IF من عبارة IF : فيمكن نفى الشرط واستبدال جزء IF مع جزء ELSE دائماً . بعد عمل ذلك ، يحتوى جزء ELSE على NEXT SENTENCE والذى يمكن اهماله ( انظر مثال ٦ – ٧ ) .

بتطبيق هذا المبدأ على مثال ٧ – ٧ نحصل على :

```
IF AMOUNT-SOLD IS NOT GREATER THAN SALES-QUOTA
    MOVE "YES" TO SEND-MOTIVATING-LETTER-SWITCH
    SUBTRACT AMOUNT-SOLD FROM SALES-QUOTA GIVING AMOUNT-UNDER-QUOTA

WRITE REPORT-LINE
    FROM WS-SALES-PERFORMANCE-LINE
```

هناك خلل بسيط فى عبارة IF – ومكافئها ، مثال ٧ – ٧ ، وهو انه ينتج خطاب تحريك ، يرسل عندما تساوى الكمية الأقل من الحصه صفرا . وتصحيح ذلك سهل :

```
IF AMOUNT-SOLD IS LESS THAN SALES-QUOTA
```

## ٧ – ٥ هيكل الاختيار : شرط الفئة

يمكن استخدام شرط الفئة class condition فى تحديد ما – اذا كان عنصر البيانات يحتوى على بيانات حرفية أو عددية . وتكوين هذا الشرط هو ما يلى :

identifier IS [NOT] { NUMERIC / ALPHABETIC }

يمكن تطبيق اختبار NUMERIC واختبار NOT NUMERIC على عناصر بيانات تكون صورها X أو SV9 فقط، ويكون استخدامها DISPLAY ( أو COMP - 3 إذا كان COMP - 3 متاحا ) . يمكن تطبيق اختبار ALPHABETIC واختبار NOT ALPHABETIC على عناصر بيانات ، تكون صورها X أو A فقط واستخدامها DISPLAY .

**الشرط identifier IS NUMERIC**

يكون صحيحا اذا كان ، واذا فقط ، احتوى العنصر على أرقام من 0 الى 9 . إذا شملت الصورةPIC اشاره عمليات (S) ، فيجب أن يحتوى العنصر كذلك على اشاره عمليات صحيحه . فإذا لم توجد اشاره فى الصوره ، فيجب ألا يحتوى العنصر نفسه على إشارة عمليات .

**ويكون الشرط : identifier IS ALPHABETIC**

صحيحا إذا ، واذا فقط ، احتوى العنصر على حروف من A إلى Z وفراغات فقط .

الاستخدام الاساسى لاختبار الفئة يكون للتأكد من صحه بيانات المدخلات فى السجلات التي يتم إدخالها بواسطه الشخص المُشغل . والبرنامج الذى يجرى الاختبار يجب أن ينتج رسائل خطأ وصفية بها معلومات كافية للسماح بتصحيح الخطأ.

مثال ٧ – ٩

```
IF PARTIAL-PAYMENT-AMOUNT  NUMERIC
    SUBTRACT PARTIAL-PAYMENT-AMOUNT FROM OLD-BALANCE GIVING
        NEW-BALANCE
ELSE
    MOVE OLD-BALANCE TO NEW-BALANCE
    MOVE "YES" TO UNPROCESSED-PAYMENT-SWITCH
```

حقل المدخلات PARTIAL - PAYMENT - AMOUNT يكتبه شخص مشغل ، ويتم التاكد من صحته باختبار فئة قبل استخدامه فى الحسابات. ولم يتم التاكد من صحه OLD - BALANCE ، لانه يأتى من ملف قرص يتم إنتاجه ببرنامج كمبيوتر . فإذا كان البرنامج مصمماً بطريقة مناسبة وخالياً من الأخطاء ، فيفترض فى الملفات التى ينتجها البرنامج أنها تحتوى بيانات صحيحة ( حيث يفترض أن البرنامج اختبر صحة كل البيانات التى وضعت فى الملف ) .

## ٧ -٦ هيكل الاختيار : شرط علاقة

إن تكوين الشرط العلاقى relation condition هى كما يلى :

```
                        { GREATER THAN }
                        { >            }
operand-1 IS [NOT]      { LESS THAN    }  operand-2
                        { <            }
                        { EQUAL TO     }
                        { =            }
```

باستثناء بعض الخليط الممنوع ( انظر مثال ٧ – ١٣) يمكن أن يكون كل من العامل الأول operand - 1 و العامل الثانى operand -2 عنصر بيانات ، أو ثابتاً ، أو تعبيراً حسابياً ( مثل المسموح به فى الطرف الأيمن لعبارة COMPUTE ) .

مثال ٧ – ١٠

- IF A + B ** 2 LESS THAN (A – C) * (A + B)

أى تعبير حسابى صحيح فى عبارة COMPUTE يكون صحيحا فى شرط العلاقة كذلك .

- IF EMPLOYEE-ID-NUMBER EQUAL HIGH-VALUES

عادة ما تستخدم الثوابت الاستعارية فى العلاقات .

- IF GROSS-PAY GREATER THAN 900.00

استخدام ثابت عددى فى شرط علاقة .

```
•    IF INPUT-RECORD-CODE = "A"
```

فى الكوبل .. يجب التعبير عن العلاقات « أقل من أو يساوى ≤ » و « أكبر أو يساوى ≥ » بأنها نفى العلاقة العكسية .

```
•    IF INPUT-RECORD-CODE NOT EQUAL "C"
```

IS و TO هى كلمات اختيارية فى شرط العلاقة .

فى الكوبل ، ماذا يعنى القول إن أحد العناصر أقل من عنصر آخر ؟

**مقارنات عددية** : لا يؤثر الاستخدام USAGE ، وحجم الصورة PICTURE على نتائج المقارنة ، والتى تكون مبنية تماما على القيم الجبرية للعناصر . تضبط العلامات العشريه تلقائيا لأغراض المقارنة . ( نفس القواعد التى تحسن من كفاءة الحسابات تحسن كذلك من كفاءه المقارنات : أنظر القسم 18 من الفصل السادس ) .

**مثال ٧ – ١١ معرفة ما يلى :**

```
01  SAMPLE-DATA-ITEMS.
    05  A     PIC S9999V99    COMP-3     VALUE  +25.67.
    05  B     PIC S99V999     COMP       VALUE  +25.670.
    05  C     PIC 99V99       DISPLAY    VALUE  25.67.
    05  D     PIC S99V99                 VALUE  -99.00.
```

تعتبر كل من A و B و C متساوية ( فهى تحتوى جميعها على نفس القيمة الجبرية 25.6 + ) D أقل من أى من C,B,A.

**مقارنات غير عددية** : هنا تعرف « أقل من » طبقا لتسلسل التتابع أثناء تنفيذ البرنامج ( انظر القسم 3 من الفصل الرابع والقسم 4 من الفصل الرابع كذلك ) . تجرى المقارنة غير العددية على بايت واحد فى نفس الوقت ، مع العمل من اليسار إلى اليمين للعنصرين المقارنين . فإذا ما اتفق العنصران على الطريق فيعتبران متساويين . وإذا حدث عدم اتفاق فى بعض المواقع ، فتتحدد النتيجة طبقا للترتيب النسبى للبايت غير المتوافق فى تسلسل التتابع . إذا اختلف طول العنصرين المقارنين ، فيعتبر الأقصر ممتدا بفراغات من ناحية اليمين .

**مثال ٧ – ١٢**

(أ) CAB أقل من CAT لأنه بالرغم من توافق أول 2 بايت ، إلا أن البايت الثالث مختلف ، B أقل من ( تسبق ) T فى تسلسل التتابع .

(ب) CAB تساوى CABbb ، حيث إن CAB ( عنصر أقصر ) يعتبر ممتدا بفراغين من ناحية اليمين لغرض المقارنة.

(جـ) COB أكبر من CAT لحدوث عدم توافق فى الموقع الثانى ، O أكبر من A فى تسلسل المقارنة . لاحظ أن B و T لن تقارنا على الإطلاق ، حيث إن النتيجة تحددت بالبايت الثانى .

(د) CAB أقل من CABBY لان CAB يعتبر ممتدا بفراغين من ناحيه اليمين لأغراض المقارنة ، ويحدث أول عدم توافق فى أول فراغ من هذين الفراغين ( CABbb تقارن مع CABBY) . وحيث إن الفراغ أقل من B فى تسلسل المقارنة ، فيكون CAB أقل من CABBY .

لقد رأينا أن المقارنات غير العدديه تحتفظ بالترتيب المعتاد لحروف الهجاء .

**مثال ٧ - ١٣ :**

كما هو الحال مع عبارة MOVE ، من الممنوع خلط بعض العوامل غير العددية والعددية فى شرط العلاقة . لاحظ ما يلى ، بصفة خاصة :

(١) يمكن مقارنة العناصر العددية المعرفة باستخدام COMP و COMP - 3 ، مع عناصر عددية فقط ( والتى يمكن أن تعرف بانها DISPLAY أو COMP و COMP - 3 ) .

(٢) لا يمكن مقارنة الثوابت أو الثوابت العددية مع بعضها البعض .

(٣) يمكن مقارنة العناصر العددية المعرفة بانها DISPLAY مع عناصر حرفية عددية PIC X - إلا أن النتائج تختلف عما هو متوقع لها . إذا فكرت فى أداء مثل هذه المقارنة ، فعليك بفحص دليل الكوبل المحدد المتاح لك .

## ٧ - ٧ هيكل الاختيار : شرط الإشارة

يمكن استخدام شرط الاشارة sign condition مع عوامل عددية فقط ، وتكوينه كما يلى :

operand IS [NOT] { POSITIVE | ZERO | NEGATIVE }

**مثال ٧ - ١٤**

- IF CURRENT-BALANCE IS POSITIVE

الشرط صحيح إذا كانت الموازنة أكبر من صفر فقط . وفيما يلى عبارة مكافئة لهذه العبارة تماما

IF CURRENT-BALANCE IS GREATER THAN ZERO

- IF CURRENT-BALANCE - PAYMENT-AMOUNT IS NOT NEGATIVE

استخدم تعبيراً حسابياً فى شرط إشارة ، وهو مكافىء لما يلى :

IF CURRENT-BALANCE - PAYMENT-AMOUNT IS NOT LESS THAN ZERO

- IF NUMBER-OVERDUE-ACCOUNTS IS ZERO

وهذه العبارة تكافىء العبارة التالية :

IF NUMBER-OVERDUE-ACCOUNTS IS EQUAL TO ZERO

- IF 12.34 IS POSITIVE

عبارة غير صحيحة : لا يمكن استخدام الثوابت أو الثوابت الاستعارية كعوامل فردية فى شرط إشارة .

- IF PERIMETER-SIZE – 12.54 IS NOT NEGATIVE

يسمح باستخدام الثوابت والثوابت الاستعارية كجزء من تعبير حسابى فى شرط إشارة .

## ٧ – ٨ هيكل الاختيار : شرط الاسم الشرطى

يمكن استخدام شرط الاسم الشرطى condition - name condition فى جزء الإجراءات كبديل مريح ووصفى لشرط العلاقة . يمكن أن يكون لاسم بيانات معين اسم شرطى condition - name واحد أو أكثر مصاحب له ، ويعرف كل من الاسماء الشرطية فى جزء البيانات بعد العنصر المقصود مباشرة ، عبر مستوى خاص وهو 88 ويكون له الشكل التالى :

88 condition-name $\left\{\begin{array}{l}\underline{\text{VALUE}}\ \text{IS}\\ \underline{\text{VALUES}}\ \text{ARE}\end{array}\right\}$ literal-1 $\left[\left\{\begin{array}{l}\underline{\text{THROUGH}}\\ \underline{\text{THRU}}\end{array}\right\}\text{literal-2}\right]$

$\left[\text{literal-3}\left[\left\{\begin{array}{l}\underline{\text{THROUGH}}\\ \underline{\text{THRU}}\end{array}\right\}\text{literal-4}\right]\right]\ldots$

يعبر عند ذلك عن شرط الاسم الشرطى كما يلى : يكون الاسم الشرطى صحيحا true إذا كان ، وإذا كان فقط ، اسم البيانات المقصود محتويا على القيمة المحددة بواسطة جزء VALUE IS .

**مثال ٧ – ١٥ :**

دعنا نعتبر أداء نشاط معين (أ) بدون (ب) وباستخدام شروط اسماء شرطية .

(أ)

```
01  CUSTOMER-UPDATE-RECORD.
    05  CUSTOMER-UPDATE-CODE          PIC X.
    05  UPDATE-CUSTOMER-NAME          PIC X(20).
    05  UPDATE-CUSTOMER-ADDRESS       PIC X(30).
 . . . . . . . . . . . . . . . . . . . . . . . .
01  WS-END-OF-FILE-SW                 PIC X(3).
 . . . . . . . . . . . . . . . . . . . . . . . .
    READ CUSTOMER-UPDATE-FILE
        AT END
            MOVE "YES" TO WS-END-OF-FILE-SW
```

```
IF WS-END-OF-FILE-SW NOT EQUAL "YES"
    IF CUSTOMER-UPDATE-CODE EQUAL "A"
        PERFORM ADD-NEW-CUSTOMER-ROUTINE
    ELSE IF  CUSTOMER-UPDATE-CODE EQUAL "D"
        PERFORM DELETE-A-CUSTOMER-ROUTINE
    ELSE IF  CUSTOMER-UPDATE-CODE EQUAL "C"
        PERFORM CHANGE-NAME-ADDRESS-ROUTINE
    ELSE
        PERFORM INVALID-UPDATE-CODE-ROUTINE
```

يحتوى أول بايت من CUSTOMER - UPDATE - RECORD على رمز يحدد ما اذا كان السجل المجدد محتويا على معلومات عن عميل جديد ليضاف إلى CUSTOMER - MASTER - FILE ، أو معلومات عن عميل يحذف من الملف ، أو معلومات عن عميل يجب تغيير اسمه وعنوانه فى الملف . ويختبر هذا الرمز بسلسلة من عبارات IF ، التى تحدد المقطع المناسب لتنفيذه ؛ بهدف تشغيل سجل العمليات الجارية التالى . وهناك الملاحظتان التاليتان :

(١) تظهر قيم الرمز كثوابت فى عبارات IF ، فإذا كان مطلوبا تغيير هذه القيم ، فيجب أن يبحث المبرمج خلال جزء الاجراءات لإيجاد مواقع حدوث الثوابت.

(٢) لا يحمل "CUSTOMR - UPDATE - CODE EQUAL "C معلومات كافية عما يعنيه عندما يكون الرمز مساويا C

(ب)

```
01  CUSTOMER-UPDATE-RECORD.
    05  CUSTOMER-UPDATE-CODE              PIC X.
        88  ADD-NEW-CUSTOMER-TRANSACTION              VALUE "A".
        88  DELETE-CUSTOMER-TRANSACTION               VALUE "D".
        88  CHANGE-ADDRESS-TRANSACTION                VALUE "C".
    05  UPDATE-CUSTOMER-NAME              PIC X(20).
    05  UPDATE-CUSTOMER-ADDRESS           PIC X(30).
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
01  WS-END-OF-FILE-SW                     PIC X(3).
    88  NO-MORE-TRANSACTION-RECORDS                   VALUE "YES".
    88  TRANSACTION-RECORD-WAS-INPUT                  VALUE "NO".
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    READ CUSTOMER-UPDATE-FILE
        AT END
            MOVE "YES" TO WS-END-OF-FILE-SW

    IF TRANSACTION-RECORD-WAS-INPUT
        IF ADD-NEW-CUSTOMER-TRANSACTION
            PERFORM ADD-NEW-CUSTOMER-ROUTINE
        ELSE IF DELETE-CUSTOMER-TRANSACTION
            PERFORM DELETE-A-CUSTOMER-ROUTINE
        ELSE IF CHANGE-ADDRESS-TRANSACTION
            PERFORM CHANGE-NAME-ADDRESS-ROUTINE
        ELSE
            PERFORM INVALID-UPDATE-CODE-ROUTINE
```

جزء VALUE مطلوب لعناصر المستوى 88 ، وغير مسموح بأى أجزاء أخرى . هذا له معنى ؛ لأن الاسم الشرطى تصاحبه قيمه خاصة لعنصر البيانات المعرف الاسم الشرطى تحته .

لاحظ كيفية استبدال شروط العلاقة من (أ) بشروط أسماء شرطية . مثل استبدال :

```
IF CUSTOMER-UPDATE-CODE EQUAL "A"
```

بمكافئها الكامل التالى :

```
IF ADD-NEW-CUSTOMER-TRANSACTION
```

تتحقق الثلاث مميزات الرئيسية من استخدام شروط الأسماء الشرطية التالية :

(١) بدون استخدام ثوابت فى عبارات IF و PERFORM من جزء الاجراءات .. يكون تعديل البرنامج أسهل إذا كان معنى رموز المدخلات أو الرموز نفسها يتغير . وكل ما يتطلب تعديل هو محتويات المستوى 88 فى جزء البيانات .

(٢) يمكن أن تعد الاسماء الشرطية بحيث تكون وصفيه للشرط الذى تصفه . وعلى هذا يحمل - CHANGE - ADDRESS TRANSALTION معنى أكبر للقارىء عن "CUSTOMER - UPDATE - CODE EQUAL "C .

(٣) يمكن للاسم الشرطى أن يمثل ببساطة شرط علاقة أكثر تعقيدا .

**مثال ٧ - ١٦**

```
01  TRANSCRIPT-RECORD.
    05  TRANSCRIPT-STUDENT-ID               PIC X(9).
    05  TRANSCRIPT-YEAR-ATTENDING           PIC 9.
88          FRESHMAN                                   VALUE 1.
88          SOPHOMORE                                  VALUE 2.
88          JUNIOR                                     VALUE 3.
88          SENIOR                                     VALUE 4.
88          UNDERGRADUATE                              VALUE 1 THRU 4.
88          GRADUATE                                   VALUE 5 THRU 9.
    05  TRANSCRIPT-CREDITS-COMPLETED        PIC 99.
    05  TRANSCRIPT-GRADE-POINT-AVERAGE      PIC 9V9.
88          HIGHEST-DISTINCTION                        VALUE 3.8 THRU 4.0.
88          HIGH-DISTINCTION                           VALUE 3.5 THRU 3.7.
88          NOT-GRADUATING                             VALUE 0.0 THRU 1.9.
    05  TRANSCRIPT-STUDENT-NAME             PIC X(30).
    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

يسمح استخدام جزء THRU بتحديد مدى range قيم للاسم الشرطى . يكون الشرط UNDER - GRADUATE صحيحا إذا كانت قيمة TRANSCRIPT - YEAR - ATTENDING تقع بين 1 و 4 بما فى ذلك الحدود ... إلخ .

يفضل بعض المبرمجين كتابة المستوى 88 فى المنطقة A ( كما سبق عمل ذلك فى المثال السابق ) ، لجعله موجودا بوضوح ، ويفضل البعض الآخر ترحيله كالمعتاد ( كما فى مثال ٧ . ١٥ - الجزء ب ) .

لاحظ الآن بساطة وسهولة فهم ما يلى :

```
IF SENIOR AND HIGH-DISTINCTION
```

عن شرط العلاقة .. المناظر له التالى

```
IF TRANSCRIPT-YEAR-ATTENDING EQUAL 4   AND
    TRANSCRIPT-GRADE-POINT-AVERAGE NOT LESS THAN 3.5   AND
    TRANSCRIPT-GRADE-POINT-AVERAGE NOT GREATER THAN 3.7
```

**مثال ٧ – ١٧**: يمكن تحديد أكثر من مدى قيم واحد لاسم شرطى معين .

```
05  LOCATION-CODE        PIC XX.
    88  PHILADELPHIA              VALUE  "AA" THRU "AC"
                                         "B7"  "D5"
                                         "E3" THRU "E5"
```

بمعرفه التعريف السابق .. يمكن استخدام ما يلى :

```
IF PHILADELPHIA
```

بدلا من التعقيد التالى

```
IF      (LOCATION-CODE NOT LESS THAN "AA"
            AND NOT GREATER THAN "AC")
    OR  (LOCATION-CODE EQUAL "B7" OR "D5")
    OR  (LOCATION-CODE NOT LESS THAN "E3"
            AND NOT GREATER THAN "E5")
```

**مثال ٧ – ١٨** : بمعرفة التعريفات التالية :

```
01  END-OF-FILE-SWITCH        PIC XXX.
    88  NO-MORE-RECORDS                 VALUE "YES".
    88  MORE-RECORDS                    VALUE "NO".
```

فإننا نلاحظ خطأين يقع فيهما المبتدئون :

```
•       READ SAMPLE-FILE
            AT END
                MOVE "YES" TO NO-MORE-RECORDS
```

( فى موقع MOVE "YES" TO END - OF - FILE - SWITCH ) . الاسم الشرطى ليس هو عنصر البيانات ، إلا أنه تمثيل شرط علاقة يشمل عنصر بيانات . والعبارة التالية :

```
MOVE "YES" TO NO-MORE-RECORDS
```

تعطى نفس الإحساس تماما مثل العبارة التالية :

MOVE "YES" TO (END-OF-FILE-SWITCH IS EQUAL TO "YES")

- IF NO-MORE-RECORDS IS EQUAL TO "YES"

يكون NO - MORE - RECORDS شرط علاقة فعلا ( فى صورة موجزة ) ، وما يؤديه هو :

IF NO-MORE-RECORDS

أو

IF END-OF-FILE-SWITCH IS EQUAL TO "YES"

## ٧ - ٩ المؤثرات المنطقية والشروط المركبة

يمكن استخدام المؤثر المنطقى الأحادى NOT فى نفى شرط بسيط simple condition ( شرط فئة أو علاقة أو إشارة أو اسم شرطى ) ، بافتراض أن الشرط البسيط نفسه لا يحتوى على NOT .

**مثال ٧ - ١٩**

- IF NOT A IS LESS THAN B

الشرط المنفى هو A IS LESS THAN B ، وعلى هذا .. يمكن أن يكون ما يلى تعويضا أوضح لعبارة IF :

IF NOT (A IS LESS THAN B)

أى صيغة مكافئة منطقيا لما يلى :

IF A IS NOT LESS THAN B

تفضل هذه الصيغة الأخيرة ، الموجودة بها NOT فى شرط بسيط ، بالنسبة إلى شروط الفئة والاشارة والعلاقة .

- IF NOT RECORD-IN-BUFFER

"RECORD-IN-BUFFER" *must* be a condition-name. If RECORD-IN-BUFFER is true, NOT RECORD-IN-BUFFER is false; if RECORD-IN-BUFFER is false, NOT RECORD-IN-BUFFER is true.

بالإضافة إلى النفى يمكن دمج الشروط البسيطة ، باستخدام المؤثرات العلاقية OR و AND ؛ لتكوين شروط مركبه -com

pound conditions . القيمة الحقيقية لـ " conditions - 1 - AND conditions- 2" تعتمد على القيم الحقيقةه لـ -condi tions- 1 و conditions- 2 كما هو موضح فى شكل ٧ – ٨ .

| condition-1 | condition-2 | condition-1 AND condition-2 |
|---|---|---|
| صحيح | صحيح | صحيح |
| صحيح | خطأ | خطأ |
| خطأ | صحيح | خطأ |
| خطأ | خطأ | خطأ |

شكل (٧ – ٨)

لاحظ إنه اذا ما ارتبط شرطان بالمؤثر AND .. فيكون الشرط المركب الناتج عن ذلك صحيحا ، عندما يكون كل من الشرطين البسيطين الأصليين صحيحا فقط .

عندما يرتبط شرطان مع بعضهما باستخدام المؤثر OR ، فيكون للشرط المركب جدول الحقيقة الموجود فى شكل ٧ – ٩.

| condition - 1 | condition - 2 | condition - 1 OR condition - 2 |
|---|---|---|
| صحيح | صحيح | صحيح |
| صحيح | خطأ | صحيح |
| خطأ | صحيح | صحيح |
| خطأ | خطأ | خطأ |

شكل ( ٧ – ٩ )

لاحظ أن الشرط المركب يكون خطأ إذا كان كل من الشرطين البسيطين خطأ فقط .

مثال ٧ – ٢٠

- IF HOURS-WORKED NUMERIC AND HOURS-WORKED GREATER THAN 40.0

الشرط المركب يكون صحيحا إذا كان ، وإذا كان فقط ، الشرطان البسيطان صحيحين . يمكن أن يحدث هذا الشرط الخاص مشاكل، ويتم تقويم شروط العلاقات فى كوبل IBM OS/ VS قبل شروط الفئات . وعلى هذا .. كان HOURS WORKED غير عددى ، يمكن أن يتسبب تقويم HOURS - WORKED GREATER THAN 40.0 فى إنهاء البرنامج نهاية غير عادية . ويمكن حذف هذه الصعوبة باستخدام عبارات IF :

```
IF HOURS-WORKED NUMERIC
    IF HOURS-WORKED GREATER THAN 40.0
```

الآن .. تم تقويم شرط العلاقة عندما كان HOURS - WORKED عدديا فقط .

- ```
  IF HOURS-WORKED NOT NUMERIC OR HOURLY-RATE NOT NUMERIC
      PERFORM ERROR-ROUTINE
  ELSE
      PERFORM CALCULATE-PAY
  ```

اذا كان أحد الشروط البسيطة صحيحا على الأقل ..فإن الشرط المركب يكون صحيحا . وعلى هذا .. ينفذ المقطع ER-ROR - ROUTINE اذا كان أى من العنصرين أو كلاهما ، غير عددى HOURS WORKED و HOURLY - RATE ، أما إذا كان العنصران عددين فينفذ CALCULATE - PAY .

- ```
  IF HOURS-WORKED NUMERIC AND HOURLY-RATE NUMERIC
      PERFORM CALCULATE-PAY
  ELSE
      PERFORM ERROR-ROUTINE
  ```

وهو مكافىء تماما لما سبق . يجب أن تتحقق ( بواسطة جداول الحقيقة ) أن العكس ( أى النفى ) لـ A OR B هو (NOT A) AND (NOT B) ، وأن عكس A AND B هو (NOT A) OR (NOT B) .

- ```
  IF RECORD-WAS-INPUT AND BALANCE DUE IS POSITIVE
  ```

شرط الاسم الشرطى هنا ، وشرط الإشارة مرتبطان بالمؤثر AND .

- ```
  IF ON-HAND NOT LESS THAN 1 OR ON-HAND NOT GREATER THAN 500
      PERFORM ADJUST-QUANTITY-ONHAND
  ```

كتابه خاطئة . الهدف كان ( كما نتوقع ) تنفيذ PERFORM ADJUST - QUANTITY - ON - HAND ، اذا لم تقع محتويات ON - HAND بين 1 أو 500 مع شمول الحدين . إلا انه بشىء من التأمل .. نرى أن الشرط المركب صحيح دائما (الطريقة الوحيدة التى يمكن أن يكون بها خطأ ، هى أن يكون ON - HAND أقل من 1 وأكبر من 500 فى نفس الوقت – وهذا أمر مستحيل ) ، وعلى هذا .. تنفذ PERFORM دائما . الشفرة الصحيحة هى :

```
IF ON-HAND LESS THAN 1 OR ON-HAND GREATER THAN 500
    PERFORM ADJUST-QUANTITY-ONHAND
```

يمكن استخدام أكثر من مؤثر منطقى واحد لعمل شرط مركب . عندما يتكرر نفس المؤثر المنطقى.. تكون الأقواس ضرورية فى الشرط المركب ( حيث تكون النتائج هى نفسها دائما ، بغض النظر عن ترتيب تطبيقات المؤثر الفردى ) . وعندما تستخدم مؤثرات منطقيه مختلفه .. فيجب أن يكون ترتيب تقويمها محددا بوضوح بواسطه الأقواس .

**مثال ٧ - ٢١**

```
•      IF      INPUT-YEAR IS GREATER THAN 80
         AND   INPUT-YEAR IS LESS THAN 84
         AND   INPUT-MONTH IS NOT LESS THAN 1
         AND   INPUT-MONTH IS NOT GREATER THAN 12
         AND   INPUT-DAY IS NOT LESS THAN 1
         AND   INPUT-DAY IS NOT GREATER THAN 31
             PERFORM PROCESS-VALID-DATE
       ELSE
           PERFORM PROCESS-INVALID-DATE
```

لاحظ أن كل شرط بسيط مكتوب فى سطر خاص به ، وأن الشروط والمؤثرات مضبوطة لتسهيل القراءة . كل الستة شروط البسيطة تكون صحيحة ؛ لكى يكون الشرط المركب صحيحا . وعلى هذا .. لكى تنفذ - PERFORM PROCESS - VALID DATE ، يجب أن تقع السنة بين 83,81 مع شمول الحدين ، ويجب أن يقع الشهر بين 1 و 12 ، ويجب ان يقع اليوم بين 1 و 31 . بينما يكتشف هذا الاختبار معظم الأخطاء فى حقل التاريخ ، فهو يترك بعض الأخطاء تمر ( مثل 83 / 31 / 09 )

```
•      IF    TIME-CARD-EMPLOYEE-NUMBER NOT NUMERIC
        OR TIME-CARD-REGULAR-HOURS NOT NUMERIC
        OR TIME-CARD-OVERTIME-HOURS NOT NUMERIC
              MOVE "YES" TO INVALID-NUMERIC-FIELDS-SW
```

يتأكد هذا المقطع من صحة حقول عددية فى TIME - CARD - RECORD لاحظ كيف يكتب كل شرط بسيط على سطر خاص به ، مع ضبط المؤثرات لتسهيل القراءة . وحيث إن OR هو المستخدم .. فإن الشرط المركب يكون صحيحا اذا كان أحد الشروط البسيطة أو أكثر صحيحا .

```
•      IF        (NOT SPECIAL-TERMS-REQUESTED)
          AND (CREDIT-UNSATISFACTORY
                 OR PURCHASE-AMOUNT GREATER THAN CREDIT-LIMIT)
```

يجب أن يكون SPECIAL - TERMS - REQUESTED , CREDIT - UNSATISFACTORY أسماء شرطية . ترتيب التقويم ، هو : (١) التقويم SPECIAL - TERMS - REQUESTED . (٢) تنفى نتيجة (١) . (٣) تقويم PURCHASE - AMOUNT GREATER THAN CREDIT - LIMIT (٤) تقويم CREDIT - UNSATISFACTORY . (٥) تربط نتيجة (٣) ونتيجة (٤) بواسطة OR . (٦) تربط نتيجة (٢) ونتيجة (٥) بواسطة AND.

```
•   IF         (NEW-CUSTOMER
          AND (CUSTOMER-ID-ON-FILE OR
               CUSTOMER-ID NOT NUMERIC) )
      OR
        (OLD-CUSTOMER
          AND (CUSTOMER-ID-NOT-ON-FILE
                 OR  DAYS-PAYMENT-LATE GREATER THAN 90) )
```

إذا أصبحت الشروط المركبه معقده بهذه الدرجة ، فهذه اشارة طيبة أنه يجب تجزئة البرنامج ، أو يجب إعادة تصميمه .

إذا لم تستخدم الاقواس ( أو اذا كانت الشروط فى نفس مستوى الاقواس ) ، فيتم تقويم التعبيرات المنطقية بالترتيب التالى : (١) التعبيرات الحسابية التى تظهر فى شروط العلاقات . (٢) الشروط البسيطة بالترتيب : العلاقة فالفئة فالاسم الشرطى فالإشاره (٣) الشروط البسيطه المنفيه بنفس الترتيب السابق ذكره . (٤) AND (٥) OR . إذا كان هناك شرطان جزئيان متساويان طبقا لترتيبهما ، فيتم تقويمهما من اليسار لليمين .

إذا بدت القواعد السابقة معقدة ، فهناك طريقة بسيطة لحذف أبعد امكانية لخطأ بسبب التقويم غير الصحيح لتعبير منطقى مركب : استخدم الاقواس .

**مثال ٧ - ٢٢ :**

بدون أقواس ، يقرأ ثالث شرط مركب فى مثال ٧ - ٢١ على النحو التالى :

```
IF NOT SPECIAL-TERMS-REQUESTED
   AND CREDIT-UNSATISFACTORY
   OR   PURCHASE-AMOUNT GREATER THAN CREDIT-LIMIT
```

ترتيب التقويم هو كما يلى : (١) شرط العلاقة PURCHASE - AMOUNT GREATER THAN CREDIT - LIMIT .

(٢) شرط اسم شرطى ، من اليسار لليمين SPECIAL - TERMS - REQUESTED .

(٣) شرط اسم شرطى ، من اليسار لليمين CREDIT - UNSATISFACTORY .

(٤) نفى شرط بسيط NOT لرقم (٢) .

(٥) ربط (٣) و (٤) بالمؤثر AND ( AND تنفذ قبل OR )

(٦) ربط (١) و (٥) بواسطة OR .

والنتيجة هى كما لو استخدمت الأقواس على النحو التالى :

```
IF         ( (NOT SPECIAL-TERMS-REQUESTED)
   AND      CREDIT-UNSATISFACTORY)
   OR      (PURCHASE-AMOUNT GREATER THAN CREDIT-LIMIT)
```

وعلى هذا .. تكون الاقواس فى المثال ٧ – ٢١ ضرورية لتحقيق النتيجة المرجوة ( والتى تختلف عن النتيجة المذكورة عاليه).

## ٧ - ١٠ اختصار شروط علاقات مركبة

شرط العلاقة البسيط العام له الشكل التالى : subject relation - operator object

يدمج شرط العلاقه المركب شرطى علاقات مركبين أو أكثر بواسطة المؤثرات العلائقية AND و OR .

subject relation - operator object { AND / OR }
subject relation - operator object

إذا كان المؤثر العلاقى relational - operator هو نفسه لشرطى علاقات بسيطين فى شرط علاقة مركب ، فلا تكون هناك حاجة لتكرار المؤثر العلاقى . وبالمثل .. إذا كان subject هو نفسه فى عديد من العلاقات البسيطة لعلاقة مركبة ، فلا يكون هناك حاجة إلى تكراره . إلا أن object يجب ان تكتب دائما لكل شرط علاقة بسيط .

مثال ٧ – ٢٣ :

- IF A EQUAL B OR A EQUAL C OR A EQUAL D

يتكرر فيها كل من subject ( عنصر البيانات A ) والمؤثر العلاقى (EQUAL ) . ويمكن حذف التكرار لنحصل على ما يلى:

IF A EQUAL B OR C OR D

بالطبع .. يجب أن يحدد أول ظهور لـ subject والمؤثر العلاقى .

- IF A EQUAL B AND A EQUAL C OR A GREATER THAN D

يمكن اختصارها لتأخذ الشكل التالى :

IF A EQUAL B AND C OR GREATER THAN D

إذا لم يوجد subject أو المؤثر العلاقى فى شرط العلاقة البسيط ، فيكون آخر subject أو آخر مؤثر علاقى يظهر فى الشرط المركب مفهوما .

- IF SAMPLE-ITEM NOT EQUAL 10.5 OR OTHER-ITEM OR 3.0

هى اختصار لما يلى :

```
IF SAMPLE-ITEM NOT EQUAL 10.5
    OR SAMPLE-ITEM NOT EQUAL OTHER-ITEM
    OR SAMPLE-ITEM NOT EQUAL 3.0
```

يجب أن يستخدم الشرط العلاقى المركب المختصر عندما لا يغيب الوضوح فقط ، بل عندما يكون هناك شك ، وعندئذ استخدم الاقواس ولا تختصر الشروط .

## ٧ - ١١ عبارة اذا المباشرة والمتداخلة

الخوارزمى المرتب المعد عن طريق دمج هياكل تتابع واختيار وتكرار عادة ما يحتوى على هياكل اختيار داخل هياكل اختيار . ونقول إن تنفيذ الكوبل الذى له مثل هذا الخوارزمى بانه يحتوى على عبارات إذا متداخلة nested IF . عندما تستخدم إذا المتداخلة يجب أن يعكس الترحيل العلاقات الموجودة بين عبارات IF بدقة.

مثال ٧ - ٢٤ :

```
IF     ADD-NEW-MASTER-RECORD
       IF MASTER-ALREADY-EXISTS
           PERFORM DUPLICATE-RECORD-ERROR
ELSE
           PERFORM CREATE-NEW-MASTER-RECORD
ELSE
       IF CHANGE-EXISTING-MASTER-RECORD
           IF MASTER-ALREADY-EXISTS
               PERFORM MAKE-CHANGES-TO-MASTER
           ELSE
               PERFORM RECORD-NOT-FOUND-ERROR
```

لاحظ كيف تم ترحيل كل عبارة IF ، وكيف ضبطت أجزاء ELSE مع IF المناظرة لها دائما . لا توجد بصفة عامة أى مشكلة فى تفسير IF متداخله ، عندما يصاحب كل IF جزء ELSE إلا أن هذا ليس لازما؛ عبارة IF CHANGE - EXIST-ING - MASTER - RECORD ليس لها جزء ELSE .

مثال ٧ - ٢٥

عند تفسير عبارة IF متداخلة ، يرفق مترجم الكوبل كل جزء ELSE مع أول IF تسبقه ، ولم يحدد لها جزء ELSE فعلا ( إذا لم تتواجد IF غير مزدوجة مع ELSE فيحدث خطأ تكوينى ) . و يمكن أن تتسبب هذه القاعدة فى حدوث مشاكل ، عندما تكتب عبارات IF المتداخله مع حذف أجزاء ELSE :

```
IF TRANSACTION-WAS-READ
    IF PAYMENT-AMOUNT NUMERIC
        SUBTRACT PAYMENT-AMOUNT FROM CURRENT-BALANCE
ELSE
    MOVE "FINAL" TO TOTAL-LINE-HEADING
    PERFORM PRODUCE-FINAL-TOTAL-LINE
```

هنا .. يرفق المترجم ELSE مع IF PAYMENT - AMOUNT ( بدلا من ارفاقها مع IF TRANSACTION - WAS - READ ، كما تبدو للمبرمج انه لديه ما يريد ) . وعلى هذا .. تنفذ العبارات كما يلى :

```
IF TRANSACTION-WAS-READ
    IF PAYMENT-AMOUNT NUMERIC
        SUBTRACT PAYMENT-AMOUNT FROM CURRENT-BALANCE
    ELSE
        MOVE "FINAL" TO TOTAL-LINE-HEADING
        PERFORM PRODUCE-FINAL-TOTAL-LINE
```

عبارة IF المتداخلة السابقة غير صحيحة ، حيث إن سطر الإجمالى النهائى ينتج فى أى وقت ، يتم إدخال دفعة غير عددية كجزء من سجل العمليات الجارية . كان الهدف الأساسى طباعة سطر الإجمالى النهائى عندما لا يمكن أن تتواجد سجلات عمليات جارية أخرى ( أى نهاية الملف ) .

لمنع مثل هذه الاخطاء .. تتطلب بعض العمليات الجاريه كتابة كل عبارة IF متداخلة يصاحبها جزء ELSE ( حتى إذا لم تكن إلا ELSE NEXT SENTENCE ) :

```
IF TRANSACTION-WAS-READ
    IF PAYMENT-AMOUNT NUMERIC
        SUBTRACT PAYMENT-AMOUNT FROM CURRENT-BALANCE
    ELSE
        NEXT SENTENCE
ELSE
    MOVE "FINAL" TO TOTAL-LINE-HEADING
    PERFORM PRODUCE-FINAL-TOTAL-LINE
```

إذا نفذت ELSE NEXT SENTENCE .. فانها تأخذ الكمبيوتر إلى العباره التى تلى النقطة .

أحيانا .. يجب أن يكرر البرنامج اختبار نفس الحقل ليحدد أى من سلسلة قيم يحتويها . و يمكن استخدام صيغة خاصة من عبارة IF المتداخله ..تسمى IF الخطية linear فى مثل هذه الحالة .

**مثال ٧ - ٢٦ :**

```
IF TRANSACTION-CODE EQUAL 1
    PERFORM CREATE-NEW-MASTER-RECORD
ELSE
    IF TRANSACTION-CODE EQUAL 2
        PERFORM DELETE-MASTER-RECORD
    ELSE
        IF TRANSACTION-CODE EQUAL 3
            PERFORM CHANGE-EXISTING-MASTER-RECORD
        ELSE
            PERFORM INVALID-TRANSACTION-CODE-ROUTINE
```

يفضل كتابة سلسلة اختبارات TRANSACTION - CODE على صورة IF خطية كما يلى :

```
IF TRANSACTION-CODE EQUAL 1
    PERFORM CREATE-NEW-MASTER-RECORD
ELSE IF TRANSACTION-CODE EQUAL 2
    PERFORM DELETE-MASTER-RECORD
ELSE IF TRANSACTION-CODE EQUAL 3
    PERFORM CHANGE-EXISTING-MASTER-RECORD
ELSE
    PERFORM INVALID-TRANSACTION-CODE-ROUTINE
```

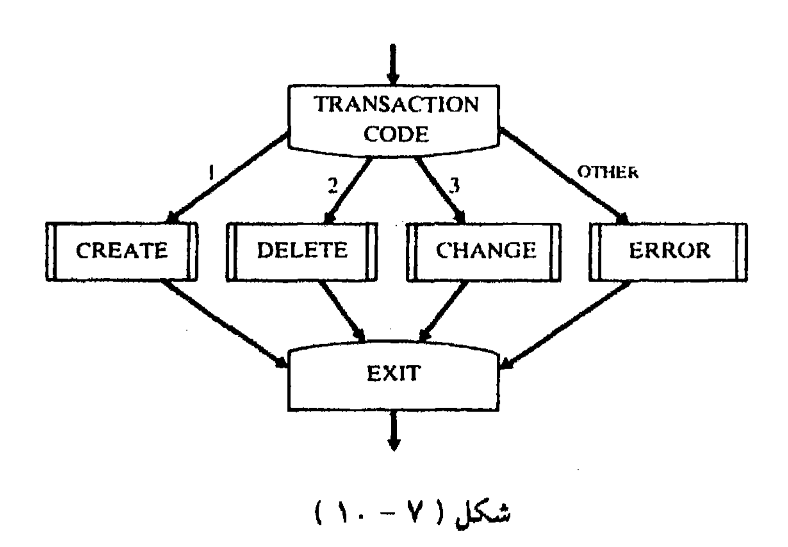
فى كل IF خطية .. تكتب كل ELSE تالية مضبوطة تحت IF الأصليه وتكتب كل IF تالية على نفس السطر الخاص بـ ELSE السابقة ، و ترحل الإجراءات كالعادة . لاحظ أن كل جزء ELSE لا تتبعه IF . العبارة ( العبارات ) المصاحبة لأخر ELSE تنفذ إذا كان عنصر البيانات الذى يجرى عليه الاختبار لا يحتوى على أى قيمة من القيم المختبرة فى شروط العلاقة السابقة فقط . لاحظ كذلك انه ينفذ مقطعاً واحداً ، ومقطعاً واحداً فقط ، الذى يعتمد على محتويات TRANSACTION - CODE فقط .

**مثال ٧ - ٢٧ :**

يمكن استخدام IF الخطية فى كتابة هيكل رابع لمنطقة البرنامج وهو هيكل الحالة ( مكون الحالة state structur ) . وهيكل الحالة ليس ضروريا على الإطلاق ، إلا أنه متاح فى بعض اللغات كوسيلة مريحة لعمل البرنامج . يتسبب هيكل الحالة فى تنفيذ أحد المقاطع بالضبط طبقا لقيمة عنصر بيانات التحكم الذى يختبر داخل الهيكل . ويمكن رسم خريطة مسار لهيكل الحالة بعدة طرق ، التى يوضح شكل (٧ - ٧) إحداها ، وهى خريطة مسار للمثال رقم ٧ - ٢٦ .

## ٧ - ١٢ هيكل التكرار : عبارة التنفيذ

تسمح عبارة التنفيذ PERFORM بتنفيذ مقطع أو قسم من جزء الإجراءات ، صفر من المرات أو مرة واحدة ، أو أى عدد من المرات . ولها أربعة أشكال .




شكل ( ٧ - ١٠ )

## تنفيذ بسيط

أبسط صيغة لعبارة PERFORM هى :

$$\underline{\text{PERFORM}}\ \text{procedure-name-1}\ \left[\left\{\begin{matrix}\underline{\text{THROUGH}}\\ \underline{\text{THRU}}\end{matrix}\right\}\ \text{procedure-name-2}\right]$$

كل اسم إجراءات procedure - name يجب أن يكون اسم مقطع أو اسم قسم .

**مثال ٧ - ٢٨ :**

```
     PERFORM  SAMPLE-PARAGRAPH
     COMPUTE...
     ........................
SAMPLE-PARAGRAPH.
     ADD...
     MOVE...
     SUBTRACT...

NEXT-PARAGRAPH.
     OPEN...
     MOVE...
```

تتسبب عبارة PERFORM فى تنفيذ كل العبارات الموجودة فى SAMPLE - PARAGRAPH فى صورة متتابعة معتادة .. بعد تنفيذ SUBTRACT .. ينتقل الكمبيوتر إلى العباره التى تلى عبارة PERFORM مباشرة .

عندما ينفذ قسم ( والذى يمكن أن يحتوى على عديد من المقاطع ) ، فتنفذ العبارات بدءا بأول عبارة ، فى أول مقطع وينتهى التنفيذ بآخر عبارة فى آخر مقطع.

**مثال ٧ - ٢٩ :**

بمعرفة المقطع المعرف فى مثال ٧ - ٢٨ تتسبب عبارة التنفيذ التالية :

```
PERFORM  SAMPLE-PARAGRAPH  THRU  NEXT-PARAGRAPH
COMPUTE...
........................................
```

فى أن يبدأ الكمبيوتر تنفيذ العبارات الموجودة فى SAMPLE - PARAGRAPH فى صورة متتابعة ، بدءا بعبارة ADD يستمر التنفيذ ، عبارة بعد أخرى ، ومقطعاً تلو الآخر ، حتى تنفذ آخر عبارة فى مقطع THRU ( عبارة MOVE) ،

وعندها يستمر تنفيذ الكمبيوتر للعبارات فى تتابع معتاد ، بدءا بالعباره التى تلى عبارة PERFORM مباشرة . (COMPUTE).

عندما يستخدم جزء THRU مع أقسام .. يبدأ التنفيذ بأول عبارة ( فى أول مقطع) من أول قسم ، وينتهى بأخر عبارة (فى أخر مقطع) من أخر قسم .

وبصفة عامة .. فإنه من الأفضل تجنب استخدام جزء THRU فى أى صيغة من صيغ PERFORM ؛ حيث إن تنفيذ PERFORM .. THRU ... يعتمد على الموقع الطبيعى فى برنامج المصدر ؛ إذ توجد إمكانية بأنه يدخل مبرمج الصيانة مقطعا أو قسما جديدا فى مدى PERFORM .. THRU عن طريق الخطأ . كما أن استخدام PERFORM .. THRU يجعل برنامج المصدر أكثر صعوبة فى فهمه ، حيث يجب أن نتذكر مواقع بداية ونهاية PERFORM . وهناك مواقف قليلة نسبيا فى برمجة الكوبل ، تدعو الحاجة فيها إلى استخدام PERFORM .. THRU ، وعلى هذا فيجب الا يكون تجنبها مرهقا.

## صيغة عدد من المرات

الصيغة التالية صيغة لأبسط من صيغ عبارة التنفيذ هى صيغة عدد من المرات TIMES ، والتى لها الشكل التالى :

$$\underline{\text{PERFORM}}\ \text{procedure-name-1}\ \left[\begin{Bmatrix}\underline{\text{THROUGH}}\\ \underline{\text{THRU}}\end{Bmatrix}\ \text{procedure-name-2}\right]\ \begin{Bmatrix}\text{dentifier-1}\\ \text{integer-1}\end{Bmatrix}\ \underline{\text{TIMES}}$$

يلعب جزء THRU نفس الدور تماما فى تعريفه مدى من العبارات التى تنفذ ، كما يفعل فى عبارة التنفيذ البسيطة . مرة أخرى .. لا نوصى باستخدامه . عندما يستخدم identifier - 1 فيجب أن يكون عنصراً فرديا وصحيحا ( مثل PIC S9 بدون علامة عشرية ) . يحدد identifier - 1 أو integer - 1 عدد المرات التى يجب أن ينفذ فيها المدى المحدد للعبارات ، قبل أن ينتقل الكمبيوتر للعبارة التى تلى عبارة PERFORM مباشرة . إذا كان identifier - 1 أو integer-1 سالبا أو صفراً، فلا ينفذ المدى المحدد للعبارة على الإطلاق ، وينتقل الكمبيوتر إلى العبارة التاليه فورا .

**مثال ٧ – ٣٠ :**

```
05  WS-NUMBER-OF-TEMPLATES      PIC 99.
. . . . . . . . . . . . . . . . . . . . . . . . .
ACCEPT WS-NUMBER-OF-TEMPLATES FROM OPERATOR-OPTIONS-DEVICE
. . . . . . . . . . . . . . . . . . . . . . . . .
PERFORM PRINT-CHECK-ALIGNMENT-TEMPLATE
    WS-NUMBER-OF-TEMPLATES TIMES
```

هنا .. يستخدم عنصر بيانات عددى فى تحديد العدد المطلوب للتكرار . يجب أن يكون OPERATOR - OPTIONS - DEVICE معرفا فى مقطع الأسماء الخاصه من جزء الأوساط . لاحظ انه إذا أدخل مشغل الكمبيوتر صفراً فى WS - NUMBER - OF - TEMPLATES ، فلا تنفذ عبارة PERFORM المقطع PRINT - CHECK - ALIGNMENT - TEMPLATE على الإطلاق .

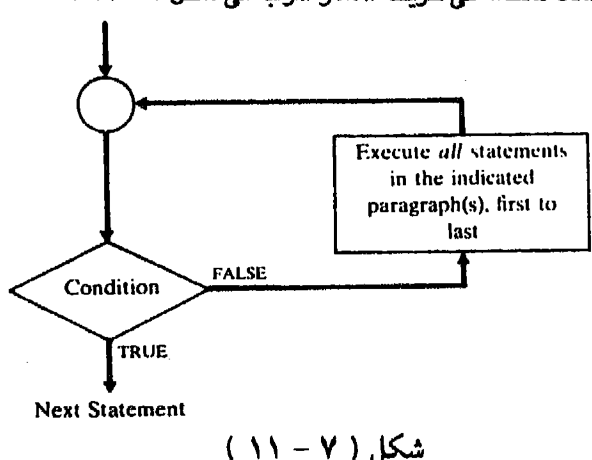
بالرغم من بساطة PERFORM ... TIMES إلا أنها لا تستخدم بكثرة فى برامج الكوبل ؛ نظرا لأنه من النادر أن يعرف عدم مرات التكرار مسبقا . الأكثر اعتيادا هو حاجتنا الى تكرار تنفيذ أحد الأجزاء حتى يتحقق شرط معين ، يحدد أن التكرار يتوقف ( مثل نهاية الملف ) .

## صيغة حتى

إن تكوين صيغة حتى UNTIL كما يلى :

$$\underline{\text{PERFORM}}\ \text{procedure-name-1}\ \left[\left\{\begin{matrix}\underline{\text{THROUGH}}\\ \underline{\text{THRU}}\end{matrix}\right\}\ \text{procedure-name-2}\right]\ \underline{\text{UNTIL}}\ \text{condition-1}$$

يوجد تاثير .. PERFORM ... UNTIL فى خريطة المسار المرتبة فى شكل ٧ - ١١ .




شكل ( ٧ - ١١ )

بالمقارنة مع شكل ٧ - ٥ : فإننا نميز هيكل DO WHILE ( مع شرط نفى ) . وعلى هذا .. كل الخواص ، التى سبق الإشارة لها فى مثال ٧ - ٣ تسرى على PERFORM ... UNTIL . ومره أخرى ..لا نوصى باستخدام جزء THRU .

مثال ٧ - ٣١ :

```
    MOVE "NO" TO NO-MORE-SWITCH
    PERFORM PRINT-A-MESSAGE
        UNTIL NO-MORE-SWITCH EQUAL "YES"
    . . . . . . . . . . . . . . . . . . . . . . . . . .
PRINT-A-MESSAGE.
    WRITE PRINT-LINE
        FROM WS-MESSAGE-LINE
```

دورة لا نهائية : بمجرد وضع NO فى NO - MORE - SWITCH .. فلن يحتوى الحقل على YES على الإطلاق . وعلى هذا .. تتكرر كتابة WRITE السجل PRINT - A - MESSAGE عدد لا نهائى من المرات . وعمليا .. تضع نظم التشغيل حدودا على طول الفتره التى يستخدم البرنامج فيها وحدة التشغيل المركزيه للكمبيوتر فيها، وعلى عدد أسطر الطباعة التى ينتجها البرنامج ، فإذا ما تعدى التنفيذ أياً من القيدين ، فيفصل البرنامج منتهياً نهاية غير طبيعية .

```
•    PERFORM DETERMINE-NEW-BALANCE
         UNTIL ALL-TRANSACTIONS-PROCESSED
```

يجب أن يكون ALL - TRANSCTIONS - PROCESSED اسماً شرطياً ، ويجب أن يصبح شرط الاسم الشرطى صحيحا true عند نقطة معينة أثناء تكرار DETERMINE - NEW - BALANCE .

```
•    PERFORM LIST-SUBASSEMBLIES
         UNTIL    ALL-ITEMS-LISTED
            OR    LINES-REMAINING IS NEGATIVE
            OR    TOTAL-ASSEMBLY-TIME IS GREATER THAN 20.0
```

يمكن استخدام أى شرط مركب فى عبارة IF و يمكن استخدامه كذلك فى عباره PERFORM ... UNTIL . هنا يتصل شرط الاسم الشرطى ALL - ITEMS - LISTED ، وشرط الإشارة ، وشرط العلاقه كلها متصله بواسطة المؤثر العلاقى OR ، لاحظ كيف كتب هذا المثال .

## صيغه مع تغيير

صيغة مع تغيير VARYING لعبارة PERFORM أكثر الصيغ تعقيدا من ناحيه التكوين ، إلا إنها عادة ماتكون مفيدة عند العمل باستخدام جداول ( انظر الفصل العاشر ) ، أو عند كتابة دورات تعيد تكرار تنفيذ المقطع ، مع تغيير محتويات عنصر بيانات معينة بطريقة يتم التحكم فيها ، فى كل مرة يحدث تكرار . ويعرض الشكل العام لعبارة PERFORM ... VARYING ... فى شكل ٧ – ١٢ ( اسماء الفهارس مشروحة فى الفصل العاشر ، فلا تهتم بها الآن ) .

```
PERFORM procedure-name-1 [ { THROUGH } procedure-name-2 ]
                           { THRU    }

VARYING { identifier-1 } FROM { literal-2      }
        { index-name-1 }      { identifier-2   }
                              { index-name-2   }

BY { literal-3    } UNTIL condition-1
   { identifier-3 }

[ AFTER { identifier-4 } FROM { literal-5    }
        { index-name-4 }      { identifier-5 }
                              { index-name-5 }

BY { literal-6    } UNTIL condition-2 ]
   { identifier-6 }

[ AFTER { identifier-7 } FROM { literal-8    }
        { index-name-7 }      { identifier-8 }
                              { index-name-8 }

BY { literal-9    } UNTIL condition-3 ]
   { identifier-9 }
```

شكل ( ٧ - ١٢ )

مثال ٧ - ٣٢ :

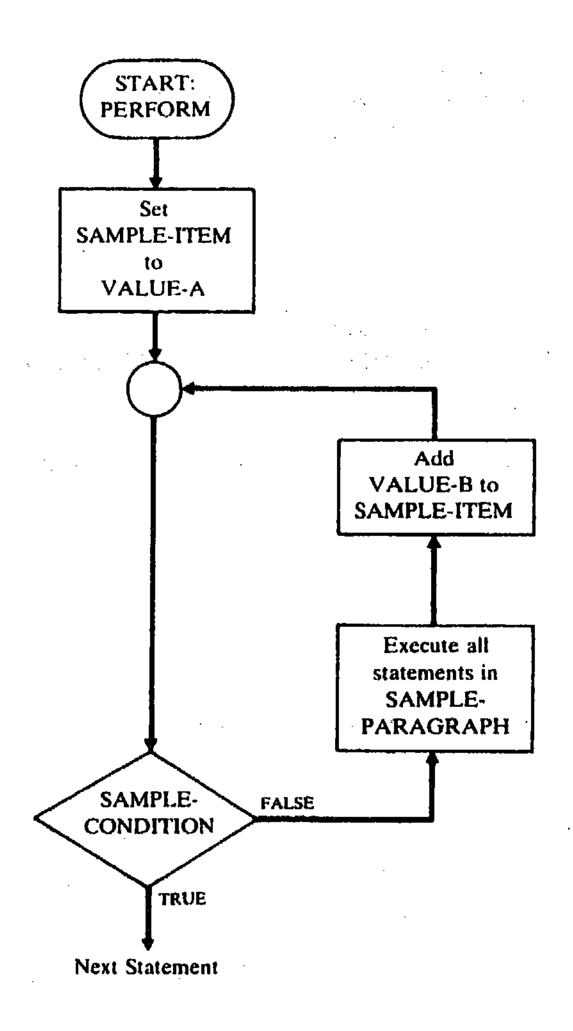
```
    PERFORM ASSIGN-MAILBOXES
        VARYING MAILBOX-NUMBER FROM 1 BY 1
        UNTIL ALL-EMPLOYEES-PROCESSED
    . . . . . . . . . . . . . . . . . . . . . . . . . . . .

ASSIGN-MAILBOXES.
    READ EMPLOYEE-MASTER-FILE
        INTO WS-EMPLOYEE-RECORD
        AT END
            MOVE "YES" TO ALL-EMPLOYEES-PROCESSED-SW

    IF ANOTHER-EMPLOYEE-INPUT
        MOVE MAILBOX-NUMBER TO WS-OUTPUT-MAILBOX-NUMBER
        MOVE WS-EMPLOYEE-NAME TO WS-OUTPUT-NAME
        WRITE MAILBOX-LIST-LINE
            FROM WS-OUTPUT-LINE
```

هذا الجزء مصمم لتحديد صناديق بريد للشركة ( مرقمة 1 , 2 , 3 , ... , ) وذلك للعاملين بالشركة . يستخدم جزء VARYING من عبارة PERFORM فى التحكم فى رقم صندوق البريد MAILBOX - NUMBER الذى يحدد للعامل الموجودة بياناته تحت التشغيل الآن . وتوضع قيمة ابتدائية للعنصر MAILBOX - NUMBER من قيمة جزء FROM وهى 1. يتم تقويم شرط UNTIL عند ذلك .. فإذا كان خطأ فينفذ المقطع ASSIGN - MAILBOXES . يقرأ مقطع ASSIGN - MAILBOXES سجل أحد العاملين ، فإذا لم يصل الى نهاية الملف ، فيطبع سطرا مبينا فيه اسم العامل والمحتويات الحاليه للعنصر MAILBOX - NUMBER . بعد تنفيذ آخر عبارة فى المقطع ASSIGN - MAILBOXES تضاف قيمة BY ، وهى 1 ، إلى MAILBOX - NUMBER لتزيد محتوياته وتصبح 2 ، عند ذلك .. يعاد تقويم شرط UNTIL ... إلخ . بعد اتمام التشغيل لآخر عامل ( وتحديد رقم صندوق بريد n له ، إذا كان عدد العاملين هو n ) ، يصبح الشرط ALL - EMPLOYEE - PROLESSED صحيحا (YES) ، ويقفز الكمبيوتر إلى أول عبارة تلى عباره PERFORM .

حدث أن تساوت قيمة FROM مع قيمه BY ، فى المثال . يبين شكل ١٢ - ٧ خريطة المسار المرتبة المناظرة لعبارة PERFORM الأكثر عمومية التالية :




شكل ( ٧ - ١٣ )

**مثال ٧ - ٣٣ :**

يمكن كتابة عبارة PERFORM التالية :

PERFORM SAMPLE - PARA 10 TIMES

كما يلي :

```
PERFORM SAMPLE-PARA
    VARYING NUMBER-OF-TIMES FROM 1 BY 1
    UNTIL NUMBER-OF-TIMES GREATER THAN 10
```

، أو كما يلي :

```
PERFORM SAMPLE-PARA
    VARYING NUMBER-OF-TIMES FROM 1 BY 1
    UNTIL NUMBER-OF-TIMES EQUAL 11
```

من الحرج أن يفهم أن ما يلى :

```
PERFORM SAMPLE-PARA
    VARYING NUMBER-OF-TIMES FROM 1 BY 1
    UNTIL NUMBER-OF-TIMES IS EQUAL TO 10
```

ينتج عنه تنفيذ SAMPLE - PARA تسع مرات فقط . السبب موضح ببساطة فى شكل ٧ - ١٣ . وبالمثل تتسبب :

```
PERFORM COMPARE-LOCATION-CODES
    VARYING LOCATION-NUMBER FROM 1 BY 1
    UNTIL LOCATION-NUMBER EQUAL TO MAXIMUM-LOCATIONS
```

فى تنفيذ COMPARE - LOCATION - CODES عدد من المرات ، أقل مرة واحدة عن المحتويات الحالية فى MAXIMUM - LOCATIONS .

مثال ٧ - ٣٤ :

```
PERFORM COMPARE-CODES
    VARYING NUMBER-COMPARES FROM VALUE-A BY VALUE-B
    UNTIL NUMBER-COMPARES GREATER THAN VALUE-C
```

لأى تغييرات فى NUMBER - COMPARES ، أو فى VALUE - B أو فى VALUE - C حدثت داخل المقطع COMPARE - CODES نفس التأثير المتوقع على عدد مرات التكرار . إلا أن تغيير قيمة FROM ( وهى VALUE - A ) أثناء تنفيذ PERFOEM لا يؤثر على عدد مرات التكرار ، حيث تلعب قيمة FROM دوراً فى بداية تنفيذ PERFORM فقط ( أى إنها تضع القيمة الابتدائية لمحتويات عنصر بيانات VARYING ).

مثال ٧ - ٣٥ :

```
PERFORM SAMPLE-PARA
    VARYING ITEM-A FROM ITEM-B BY ITEM-C
    UNTIL ITEM-A GREATER THAN ITEM-D
```

عندما تستخدم عناصر بيانات مع PERFORM ... VARYING ... يجب أن تكون كلها عناصر بيانات فردية عددية . وعلى هذا .. يجب أن تكون صورة العناصر ITEM-A , ITEM-B , ITEM-C , INEM -D بها 9 ، ويمكن أن يكون بها S ، أو V اختياريين .

```
PERFORM SAMPLE-PARA
    VARYING ITEM-A FROM .25 BY .03
    UNTIL ITEM-A GREATER THAN .40
```

ينفذ SAMPLE - PARA عدد 6 مرات ، مع أخذ العنصر ITEM-A القيم 25 ,.28, .31, .34, .37, .40. ، وينتهى تنفيذ PERFORM عندما تساوى قيمة ITEM-A 43.

```
●    PERFORM SAMPLE-PARA
         VARYING ITEM-A FROM .25 BY .03
         UNTIL ITEM-A EQUAL TO .41
```

ينتج عن عبارة PERFORM هذه دورة لا نهائية ، حيث لا يتحقق الشرط ITEM-A EQUAL TO.41 علــى الإطلاق (الرقم 25. - 41. لا يقبل القسمة على 03. ) . انظر مثال ٧ – ٣١

```
●    PERFORM SAMPLE-PARA
         VARYING ITEM-A FROM 10 BY -1
         UNTIL ITEM-A IS ZERO
```

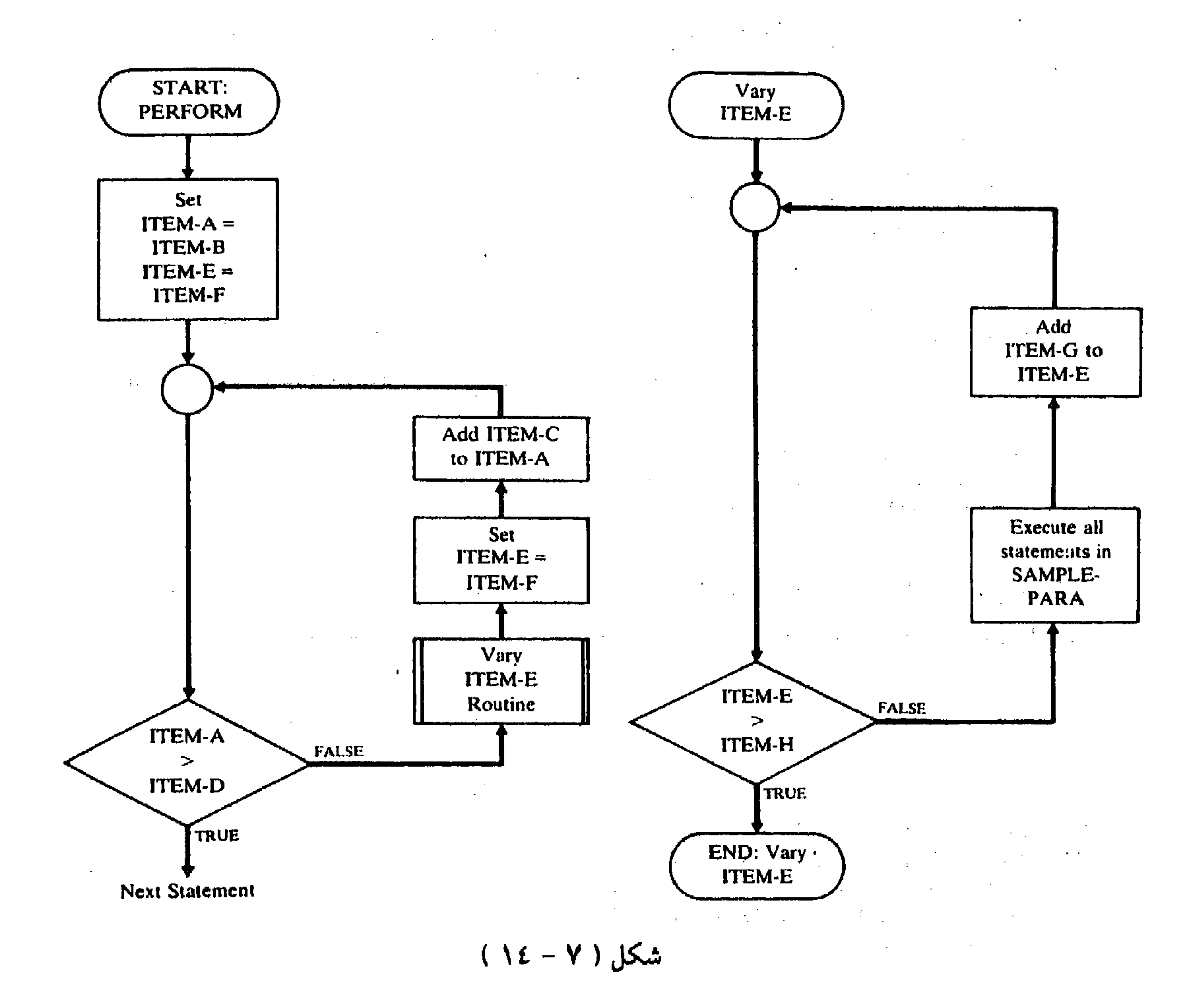
عندما تكون قيمة BY سالبة .. فيأخذ العنصر ITEM-A القيم التالية : 1,2,3,4,5,6,7,8,9,10 ، وينفذ SAMPLE - PARA عشر مرات . بعد التنفيذ للمرة العاشرة .. يساوى ITEM - A صفراً ، وينتقل الكمبيوتر الى العبارة التى تلى عبارة PERFORM مباشرة .

عندما يتغير عنصرا بيانات فى عبارة PERFORM...VARYING .

```
PERFORM SAMPLE-PARA
    VARYING ITEM-A FROM ITEM-B BY ITEM-C
        UNTIL ITEM-A GREATER THAN ITEM-D
    AFTER ITEM-E FROM ITEM-F BY ITEM-G
        UNTIL ITEM-E GREATER THAN ITEM-H
```

فيكون التنفيذ كما يبدو فى خريطة المسار المرتبة فى شكل ٧ – ١٤ . يمكن رؤيه أنه لكل قيمة من قيم ITEM - A ( التى تقع فى المدى ITEM - B ≤ ITEM - A ≤ ITEM - D ) ينفذ المقطع SAMPLE - PARA لكل قيم ITEM - E ( التى تقع فى المدى ITEM - F ≤ ITEM - E ≤ ITEM - H ) . وعلى هذا .. يكون تاثير PERFORM ... VARYING هو تنفيذ متكرر للمقطع SAMPLE - PARA مع :

إجمالى عدد مرات التكرار = (عدد القيم التى تقع فى مدى ITEM - A ) x ( عدد القيم التى تقع فى مدى ITEM - E ).




شكل ( ٧ - ١٤ )

**مثال ٧ - ٣٦ :**

```
PERFORM CALCULATE-MORTGAGE-PAYMENT
    VARYING PAYMENT-YEAR FROM 83 BY 1
        UNTIL PAYMENT-YEAR GREATER THAN 93
    AFTER PAYMENT-MONTH FROM 1 BY 1
        UNTIL PAYMENT-MONTH GREATER THAN 12
```

ينفذ CALCULATE - MORTAGE - PAYMENT لكل قيمة من قيم PAYMENT - YEAR من 83 إلى 93 ، مع شمول الحدود . ولكل من هذه القيم البالغ عددها 11 للعنصر PAYMENT - YEAR تتغير قيمة PAYMENT - MONTH من 1 الى 12 . وعلى هذا .. يحسب المقطع (11x12) 132 قيمه لدفعات القرض ، والتي يمكن أن تمثل على هيئة جدول ، أو في صورة خطية ( شكل ٧ - ١٥ ) .

| Month / Year | 1 | 2 | ... | 12 |
|---|---|---|---|---|
| 83 | $xxx | | | |
| 84 | | | | |
| ⋮ | | | | |
| 93 | | | | |

(a)

| Year | Month | Payment |
|---|---|---|
| 83 | 1 | $xxx |
| 83 | 2 | |
| . . . | . . . | . . . |
| 83 | 12 | |
| 84 | 1 | |
| 84 | 2 | |
| . . . | . . . | . . . |
| 84 | 12 | |
| : : : | : : : | : : : |
| 93 | 1 | |
| 93 | 2 | |
| . . . | . . . | . . . |
| 93 | 12 | |

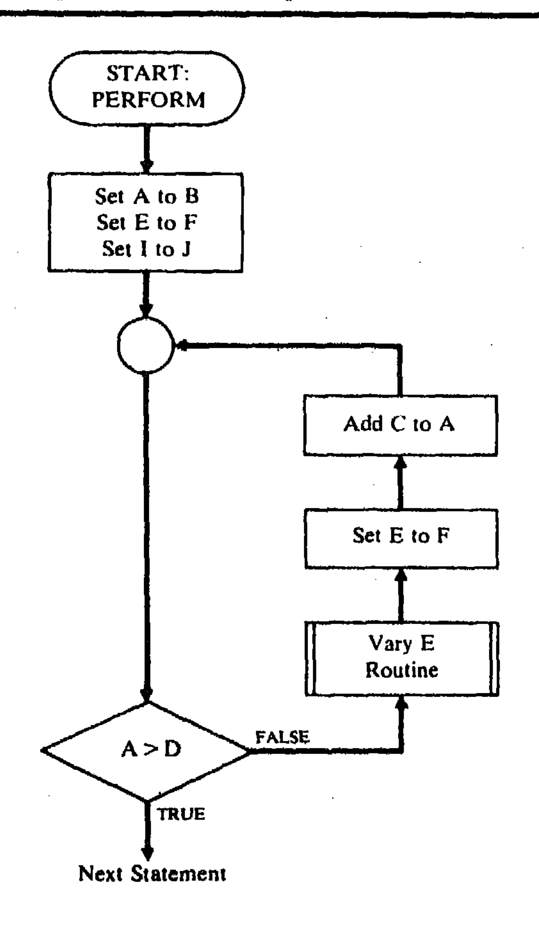
(b)

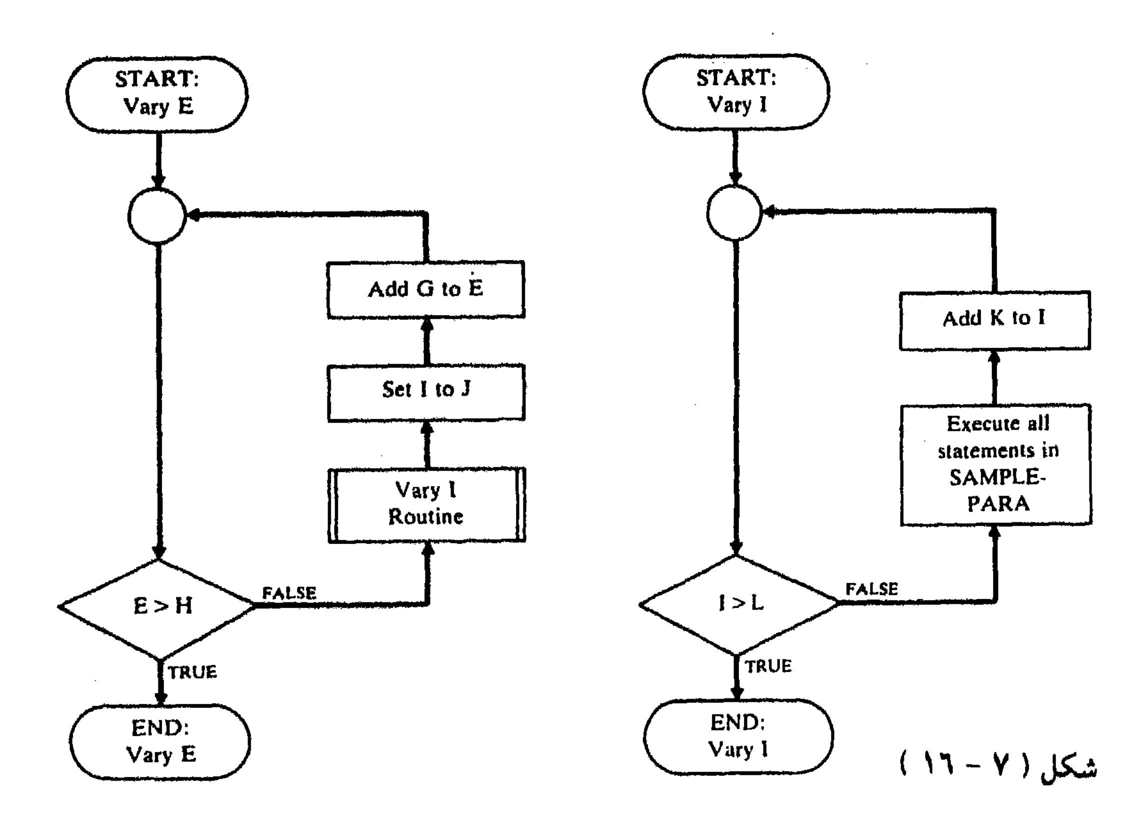
شكل ( ٧ - ١٥ )

يمكن أن يتغير أقصى رقم لعناصر بيانات 3 فى عبارة PERFORM ... VARYING (انظر ملحق حـ لمعرفة العدد المناظر فى كوبل 1985 النمطى) :

```
PERFORM SAMPLE-PARA
    VARYING A FROM B BY C
        UNTIL A GREATER THAN D
    AFTER E FROM F BY G
        UNTIL E GREATER THAN H
    AFTER I FROM J BY K
        UNTIL I GREATER THAN L
```

لكل قيمة تؤخذ للعنصر A .. تتغير قيم E خلال مداها كله ، ولكل قيمة من قيم E ، تتغير قيم I خلال مداها كله . يتبع ذلك أن عدد مرات تكرار المقطع SAMPLE - PARA هو : ( عدد قيم A ) x ( عدد قيم E ) x ( عدد قيم I ) ، ويوضح شكل (٧-١٦) خريطة المسار المرتبة لهذه الحالة .


START:
PERFORM
Set A to B
Set E to F
Set I to J
Add C to A
Set E to F
Vary E
Routine
A > D
FALSE
TRUE
Next Statement


START:
Vary E
Add G to E
Set I to J
Vary I
Routine
E > H
FALSE
TRUE
END:
Vary E
START:
Vary I
Add K to I
Execute all
statements in
SAMPLE-
PARA
I > L
FALSE
TRUE
END:
Vary I

شكل ( ٧ - ١٦ )

مثال٧ – ٣٧ :

```
PERFORM PRODUCE-TOTAL-INTEREST
    VARYING MORTGAGE-AMOUNT
        FROM 50000 BY 1000
        UNTIL MORTGAGE-AMOUNT GREATER THAN 100000
    AFTER YEARS-TO-REPAY
        FROM 10 BY 1
        UNTIL YEARS-TO-REPAY GREATER THAN 30
    AFTER INTEREST-RATE
        FROM .08 BY .01
        UNTIL INTEREST-RATE GREATER THAN .20
```

يمكن استخدام عبارة PERFORM فى إنتاج تقرير يبين إجمالى قيم الفائدة لقيم القرض المختلفة ، عند معدلات فائدة مختلفة بأطول زمنية مختلفة للسداد . يمكن أن تتغير قيم القرض من 50,000 إلى 100,000 بخطوات متساوية تساوى 1000 . لكل قيمة من قيم MORTAGE - AMOUNT .. تتغير قيمة YEARS - TO - REPAY من 10 إلى 30 بخطوات متساوية تساوى 1 .. لكل قيمة من قيم MORTAGE - AMOUNT ، ولكل قيمة من قيم YEARS - TO - REPAY تتغير قيمة INTER- EST - RATE من 8% إلى 20% بخطوات متساوية تساوى 1% . يمكن طباعه التقرير كقائمه خطية ، أو ربما من الأفضل ، كمجموعة من 51 جدولا ، يعطى كل منها إجمالى الفائدة لكل خليط لطول الفترة ومعدل الفائدة .

## ٧ - ١٣ عبارات التنفيذ المتداخلة

كما أنه يسمح صحيح استخدام عبارة IF داخل عباره IF ( القسم رقم 11 من هذا الفصل ) .. فمن الصحيح كذلك استخدام عبارة PERFORM داخل مقطع ينفذ بواسطة عبارة PERFORM أخرى .

مثال ٧ – ٣٨ :

```
    PERFORM PARA-A
    MOVE "YES" TO SAMPLE-SWITCH
    . . . . . . . . . . . . . . . . . . . . . . . . .
PARA-A.
    MOVE 1 TO L
    MOVE 2 TO M
    PERFORM PARA-B
    MOVE 3 TO N
    .
    . . . . . . . . . . . . . . . . . . . . . . . . .
PARA-B.
    ADD 1 TO L
    ADD 2 TO M
    PERFORM PARA-C
        VARYING AN-ITEM FROM 1 BY 1
        UNTIL AN-ITEM GREATER THAN 3
    ADD 3 TO P
    .
    . . . . . . . . . . . . . . . . . . . . . . . . . .
PARA-C.
    MOVE SPACES TO X
    MOVE LOW-VALUES TO Y
    .
```

يتسبب تنفيذ عبارة PERFORM فى أعلى مستوى فى تنفيذ PARA - A مرة واحدة .إلا أن المقطع PARA - A يحتوى على عباره PERFORM ؛ بحيث يؤدى تنفيذ PARA - A إلى تنفيذ للمقطع PARA - B مرة واحدة . بالمثل يحتوى المقطع PARA - B على عباره PERFORM ، بحيث ان تنفيذ PARA - B ينتج عنه تنفيذ PARA - C ثلاث مرات . تذكر انه عندما ينتهى تنفيذ عبارة PERFORM .. يعود التحكم تلقائيا إلى العبارة التى تلى عباره PERFORM مباشره . وعلى هذا ينتقل الكمبيوتر تلقائيا من ثالث تكرار لآخر عباره فى PARA - C الى العباره التى تلى عبارة PERFORM فى PARA - B. من أخر عبارة فى PARA - B .. ينتقل الكمبيوتر تلقائيا إلى العبارة التى تلى عبارة PERFORM فى PARA - A ومن آخر عباره فى PARA - A .. ينتقل الكمبيوتر إلى العبارة التى تلى عبارة PERFORM التى بدأت كل هذا التسلسل . وعلى هذا تنفذ العبارات بالترتيب التالى :

```
1.  PERFORM PARA-A
2.  MOVE 1 TO L
3.  MOVE 2 TO M
4.  PERFORM PARA-B
5.  ADD 1 TO L
6.  ADD 2 TO M
7.  PERFORM PARA-C VARYING...
8.  MOVE SPACES TO X (1st iteration; AN-ITEM = 1)
9.  MOVE LOW-VALUES TO Y (1st iteration; AN-ITEM = 1)
10. MOVE SPACES TO X (2nd iteration; AN-ITEM = 2)
11. MOVE LOW-VALUES TO Y (2nd iteration; AN-ITEM = 2)
12. MOVE SPACES TO X (3rd iteration; AN-ITEM = 3)
13. MOVE LOW-VALUES TO Y (3rd iteration; AN-ITEM = 3)
14. ADD 3 TO P
15. MOVE 3 TO N
16. MOVE "YES" TO SAMPLE-SWITCH
```

يشمل المصدر الرئيسى للخطأ عند استخدام عبارات PERFORM متداخلة كتابة عبارة PERFORM ، التى تنفذ نفسها بصورة مباشرة أو غير مباشرة . عندما يستدعى جزء من البرنامج .. فيقال عنه انه يتسم بالاعاده الذاتيه recursive. الإعاده الذاتية لعبارات PERFORM – غير مسموح بها فى الكوبل ( بالرغم من أن الإعاده الذاتية ممكنة فى لغات أخرى للبرمجه مثل PL/I ) .

مثال ٧ - ٣٩ :

```
PROCEDURE DIVISION.
    OPEN INPUT TEST-FILE
    MOVE "NO" TO END-FILE-SWITCH
    PERFORM READ-AND-VALIDATE
        UNTIL END-FILE-SWITCH EQUAL "YES"
    CLOSE TEST-FILE
    STOP RUN

READ-AND-VALIDATE.
    READ TEST-FILE
        AT END
            MOVE "YES" TO END-FILE-SWITCH

    IF END-FILE-SWITCH EQUAL "NO"
        IF TEST-RECORD-FIELD NUMERIC
            PERFORM VALID-RECORD-PROCESSING
        ELSE
            PERFORM PRODUCE-ERROR-MESSAGE
            PERFORM READ-AND-VALIDATE
```

أخر عبارة فى المقطع READ - AND - VALIDATE هي عبارة PERFORM READ - AND - VALIDATE. وعلى هذا .. تحت الظروف الصحيحة (TEST - RECORD - FIELD ليس عدديا ) تنفذ عبارة PERFORM نفسها . هذا خطأ برمجة بالقطع ، يبدو فى كوبل IBM OS/VS كدورة لا نهائية .

يفكر المبرمج حتما أنه إذا ما وجد سجلاً غير صحيح .. فإن الشيء المناسب عمله هو طباعة رسالة خطأ ، والاستمرار بقراءة السجل التالى وتشغيله . الطريقه صحيحة فى الواقع – ولكن ما سبق أن نسيه هو أن جزء READ - AND - VALIDATE يقع تحت تحكم عبارة PERFORM...UNTIL سابقة . ليس من الصحيح تنفيذ PERFORM ثانيه للمقطع ، قبل اتمام تنفيذ PERFORM الأولى . فى هذه الحالة ، تجعل PERFORM...UNTIL الأولى من PERFORM الثانية عبارة غير ضرورية بأى حال من الأحوال .

مثال ٧ - ٤٠:

```
PROCEDURE DIVISION.
. . . . . . . . . . . . . . . . . . . . . . .
PARA-A.
    MOVE 1 TO L
    MOVE 2 TO M
    PERFORM PARA-B
    MOVE 3 TO N
    .
. . . . . . . . . . . . . . . . . . . . . . .
PARA-B.
    ADD 1 TO L
    ADD 2 TO M
    PERFORM PARA-C
    ADD 3 TO P
```

```
PARA-C.
    MOVE SPACES TO X
    PERFORM PARA-A
    MOVE LOW-VALUES TO Y
```

هنا تدور العبارة PERFORM PARA - B حول تنفيذ نفسها ، ولكن هذا بطريقة غير مباشرة ، وذلك بعد سلسلة من عبارات PERFORM متداخلة . وهذا خطأ يشبه الدورة اللانهائية . يحتوى PARA - A على عبارة PERFORM تنفذ PARA - B ( هذا صحيح حتى الآن ) . ويحتوى PARA - B على عبارة PERFORM تنفذ PARA - C (مازال هذا صحيحاً) ، إلا أن PARA - C يحتوى على عبارة PERFORM تنفذ PARA - A ، و هذا غير صحيح لأنه أثناء إعادة تنفيذ PARA - A .. تنفذ العبارة PERFORM PRA - B ، بينما لا تكون عبارة PERFORM PARA - B قد نفذت بالفعل.

ولتجنب مثل هذه المواقف الموجودة فى مثال ٧ - ٤٠ فإننا نلاحظ ما يلى :

**قاعدة لعبارات PERFORMs المتداخلة ..**

المقطع أو القسم الذى ينفذ تحت تحكم عباته PERFORM يجب ألا ينفذ مرة أخرى ، حتى ينتهى تنفيذ عبارة PERFORM المتحكمة فيه أولا .

عندما يعود التحكم من الإجراء المنفذ .. فسيكون مسموحا ، بالطبع ، تنفيذ الإجراء مرة أخرى .

## ٧ - ١٤ التصميم المنطقى : الشفرة الشبيهة

عادة ما تستخدم الشفرة الشبيهة pseudocode بدلا من خرائط المسار المرتبة .. ، وذلك فى تصميم منطق البرنامج وتوثيقه. ونظرا لتشابه الشفرة الشبيهة مع اللغة الانجليزية ، فعادة ما يشار إليها بالانجليزية المرتبة structured english . وحيث إن الشفرة الشبيهة ليست رسومات .. فهى أسهل كثيرا فى إنتاجها وتعديلها عن خرائط المسار المرتبة .. (التى يجب أن ترسم بعناية).

ولا توجد صيغة موحدة للشفرة الشبيهة ، والمتطلب الوحيد هو أن الصيغة تكون قادرة على تمثيل الثلاثة هياكل الرئيسية للمنطق : تتابع واختيار وتكرار . توصف المدخلات والتشغيل والمخرجات للبيانات بعبارات إنجليزية ( موجزة ) ، ويستخدم الترحيل فى توضيح العلاقات بين عبارات الشفرة الشبيهة .

**مثال ٧ - ٤١ :**

شفرة شبيهة لهياكل المنطق الأساسية .

***Sequence***

```
        increment days overdue counter by 1
        compute late charge as 5% of balance due
        build dunning statement for printing
```

***Selection***

```
        if hours worked is greater than 40
        . . . . . . . . . . . . . . . . . . .
        else
        . . . . . . . . . . . . . . . . . . .
        endif
```

***Iteration***

```
        perform until transaction customer ID not equal master customer ID
        . . . . . . . . . . .
        endperform
```

من الممكن بالطبع نهج مناهج أخرى ، طالما أنها (١) تفهم قراءتها (٢) ومتماسكة (أى إنك إذا استخدمت perform/ end-perform للتكرار فى أحد الأماكن من الشفرة الشبيهة ، فعليك باستخدامها فى كل المواقع كذلك ) .

**مثال ٧ – ٤٢ :**

اذكر شفرة شبيهة مكافئة (أ) لشكل ٧ – ٤ (ب) لشكل ٧ – ٧

(أ)

```
read payment date, payment, due date, old balance
if payment date greater than due date
     call late payment routine
else
     if payment greater than balance
          calculate refund
          format check
          print check
     else
          compute new balance as old balance – payment
     endif
endif
build cash received line
```

لاحظ أن هذه الشفرة الشبيهة تستدعى عملية سبق تعريفها فى العبارة : call payment routine . يمكن أن يقبل كذلك استخدام perform late payment routine و execute late payment routine ، أو أى عبارة تحدد بوضوح ما يراد عمله عند هذه النقطة من الخوارزمى .

(ب)

```
set end-cards to "no"
set #-employees to zero
read card-file at end set end-cards to "yes"
perform until end-cards equal "yes"
     increment #-employees by 1
     move name, regular hours, and overtime hours to output areas
     compute total-hours as regular hours + overtime hours
     print time listing line with name, regular hours, overtime hours, and
          total hours
     read card-file at end set end-cards to "yes"
endperform
print #-employees
stop
```

وصيغة الشفرة الشبيهة أكثر إيجازا عن خريطة المسار المرتبة . لاحظ أن عبارات الشفرة الشبيهة المراد تكرارها فى هيكل تكرار ، مكتوبة بين عبارات perform until , endpenform فى ترجمة الكوبل ، العبارات المراد تكرارها بالطبع ، كمقطع منفصل ، ينفذ بواسطة PERFORM ... UNTIL (انظر ملحق حـ بالنسبة إلى كوبل 1985 النمطى) .

يفضل معظم المبرمجين الشفرة الشبيهة عن خرائط المسار المرتبة . وبأى طريقة .. يجب أن يصمم البرنامج قبل كتابة أى شفرة بلغة الكوبل . يجب كتابة جزء الإجراءات فى الواقع باستخدام الشفرة الشبيهة أو خريطة المسار المرتبة كدليل .

يمكن توفير كثير من وقت التصحيح إذا ما اختبرت الشفرة الشبيهة أو خريطة المسار المرتبة قبل بدء كتابة شفرة الكوبل . ويعنى الاختبار أن تقوم بالدور الذى يلعبه الكمبيوتر ، وتتبع الخوارزمى خطوة بخطوة ، مجريا تشغيل لبعض بيانات اختيارية ومنتجا نتائج ؛ فاذا كانت النتائج صحيحة ، فيجب أن يكون الخوارزمى صحيحا . وإذا ما تمت ترجمته ترجمة صحيحة إلى الكوبل .. فيجب أن يكون برنامج الكوبل صحيحا .

## ٧ - ١٥ التصميم المنطقى : جداول القرارات

يمكن أن تصبح الشفرة الشبيهة وخرائط المسار المرتبه مرهقة عندما يتطلب الخوارزمى هياكل اختيار معقدة (مثل التداخل العميق). جدول القرار DECISION TABLE عبارة عن رسم يلخص بطريقة مقنعة اجراءات قرارات معقدة complicated de- cision priocedures ، والأكثر دقة ، انه جدول حقيقى (القسم رقم ٩ من هذا الفصل) لمجموعة من العبارات (تسمى إجراءات actions المنطقية (المعقدة) والمركبة من مجموعة من العبارات البسيطة (تسمى شروطاً) ، وتخطيط جدول القرارات م غسح فى شكل ٧ – ١٧ .

| | |
|---|---|
| *condition stub* | *conditions* |
| *action stub* | *actions* |

شكل ( ٧ - ١٧ )

**مثال ٧ -٤٣ :**

شكل ٧ . ١٨ هو جدول قرارات يمثل شراء على الحساب من أحد المحلات . جزء الشروط condition stub يسرد ثلاثة شروط ، كل منها يمكن أن يكون صحيحاً ( نعم Y ) أو خطأ ( لا N ) .

لاحظ أن كل الخليط الممكن من الصحه والخطأ للثلاثة شروط ممثل بأعمدة شروط = $2^3$ أى 8 أعمدة .( لعدد n من الشروط يصبح عدد الأعمدة $2^n$ ) . جزء الإجراءات action stub يسرد ثلاثة إجراءات ممكنة يمكن اتخاذها بالنسبة للشراء . (بالصدفة تساوى عدد الإجراءات مع عدد الشروط هنا). فى كل عمود توجد X عند كل إجراء يجب أخذه لخليط معين من قيم الحقيقة . انظر التمرين رقم 91 فى هذا الفصل للشفرة الشبيهة لجدول القرارات هذا، وجدول القرارات أسهل كثيراً فى صياغته عن الشفرة الشبيهة).

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Account Active | Y | Y | Y | Y | N | N | N | N |
| Last Month's Payment Received | Y | Y | N | N | Y | Y | N | N |
| Purchase Is Within Credit Limit | Y | N | Y | N | Y | N | Y | N |
| Approve Purchase | × | × | × | | | | | |
| Refer Customer to Credit Office | | × | × | | | | | |
| Disapprove Purchase | | | | × | × | × | × | × |

شكل ( ٧ - ١٨ )

## أسئلة للمراجعة :

٧ – ١ ماذا يعنى منطق البرنامج ؟

٧ – ٢ ما الثلاث هياكل المنطقية الاساسية ؟

٧ – ٣ ما خريطة المسار المركبة ؟ وكيف تستخدم ؟

٧ – ٤ ما الغرض من كل رمز من رموز خريطة المسار المرتبة التى سبق ذكرها فى هذا الفصل ؟

٧ – ٥ وضح كيف ترسم خريطة المسار لهيكل التتابع .

٧ – ٦ وضح كيف ترسم خريطة المسار لهيكل الاختيار .

٧ – ٧ وضح كيف ترسم خريطة المسار لهيكل التكرار .

٧ – ٨ وضح كيف يمكن دمج الثلاثة هياكل للمناطق الثلاثة فى خريطة مسار .

٧ – ٩ ما هيكل DO WHILE ؟

٧ – ١٠ كيف ينفذ هيكل الاختيار فى الكوبل ؟

٧ – ١١ كيف يمكن استخدام NEXT SENTENCE ؟

٧ – ١٢ وضح أهميه النقطة فى عباره IF .

٧ – ١٣ وضح تكوين واستخدام شرط الفئة .

٧ – ١٤ متى يكون عنصر بيانات عدديا ؟ ومتى يكون حرفيا ؟

٧ – ١٥ أى عناصر البيانات يجب أن يتأكد برنامج الكوبل من صحتها ؟

٧ – ١٦ ماذا يحدث إذا استخدمت البيانات عددية غير صحيحة فى حساب أو فى شرط علاقة ؟

٧ – ١٧ متى يجب ألا تختبر صحة بيانات المدخلات ؟

٧ – ١٨ وضح تكوين واستخدام شرط العلاقة .

٧ – ١٩ كيف تنفذ العلاقات «أقل من أو يساوى» و« أكبر من أو يساوى » فى الكوبل ؟

٧ – ٢٠ وضح القواعد المتبعة فى مقارنة عنصرين عدديين فى الكوبل .

٧ – ٢١ كيف تقارن العناصر غير العددية فى الكوبل ؟

٧ – ٢٢ ما مجموعات خليط عناصر البيانات غير المسموح بها فى شروط العلاقات ؟

٧ – ٢٣ وضح تكوين واستخدام شرط الإشارة .

٧ – ٢٤ اذكر ثلاث مميزات لاستخدام شروط أسماء شرطية بدلا من شروط العلاقات .

٧ – ٢٥ كيف تعرف الأسماء الشرطية ؟

٧ – ٢٦ كيف يتم تقويم شرط مركب ، مكون بواسطة المؤثرات العلاقية NOT, OR, AND؟ ؟

٧ – ٢٧ متى يجب استخدام الأقواس فى الشروط المركبة ؟

٧ ـ ٢٨ ما القواعد المتبعة إذا لم توجد أقواس في الشرط المركب ؟

٧ ـ ٢٩ اذكر قواعد اختصار شروط العلاقات المركبة . متى يجب استخدام الاختصار ؟ ومتى لا يجب ؟

٧ – ٣٠ ماذا تعني IF المتداخلة ؟

٧ – ٣١ وضح كيف تتزاوج IF مع ELSE في IF المتداخلة .

٧ – ٣٢ وضح دور الترحيل عند كتابة IF متداخلة.

٧ – ٣٣ وضح استخدام NEXT SENTENCE في IF المتداخلة .

٧ – ٣٤ وضح هيكل الحالة .

٧ – ٣٥ ما هي IF الخطية ؟ وضح كيفية استخدامها في تنفيذ هيكل الحالة .

٧ – ٣٦ ناقش تكوينات الصيغ الأربعة لعبارة PERFORM .

٧ – ٣٧ وضح جزء THRU من عبارة PERFORM . لماذا يجب تجنب استخدامه ؟

٧ – ٣٨ كيف تستخدم PERFORM ... UNTIL في تنفيذ هيكل التكرار ؟

٧ – ٣٩ ماذا يحدث إذا أصبح شرط UNTIL صحيحا في منتصف تنفيذ المقطع ؟

٧ – ٤٠ ما الدورة اللانهائية ؟

٧ – ٤١ هل من الممكن تنفيذ صفر من المرات ؟ كيف ؟

٧ – ٤٢ ناقش الاختلافات بين : PERFORM ... VARYING و PERFORM ... VARYING ... AFTER ... PERFORM .. VARYING ... AFTER ...

٧ – ٤٣ ناقش الاختلاف بين EQUAL, GREATER THAN بالنسبه الى عدد مرات التكرار التى تتحقق بواسطة PERFORM ... UNTIL .

٧ – ٤٤ ماذا تعني عبارة PERFORM المتداخلة ؟

٧ – ٤٥ اذكر الخطأ الذي يمكن أن يقع من استخدام PERFORM المتداخلة .

٧ – ٤٦ ما الشفرة الشبيهة ؟ ولماذا تفضل عادة عن خريطة المسار المرتبة ؟

٧ – ٤٧ أكتب شفرة شبيهة . وارسم خريطة مسار لكل عبارة PERFORM في السؤال رقم ٤٢ .

٧ – ٤٨ ما الاختبار الذي يجرى على الشفرة أو خريطة المسار ؟

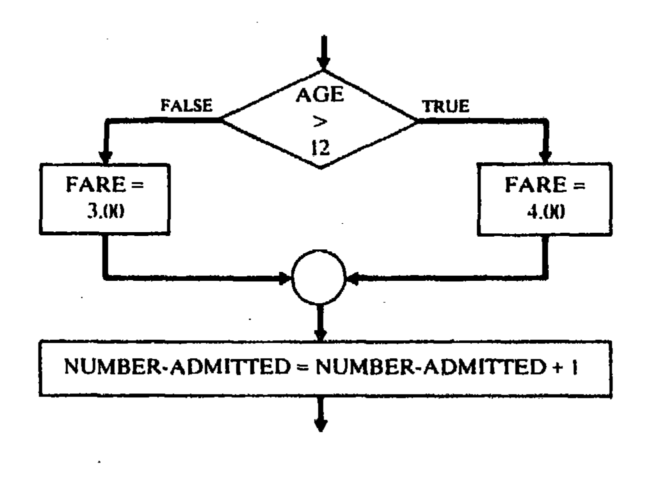
٧ – ٤٩ وضح استخدام جداول القرارات . كيف ترسم هذه الجداول ؟

## مسائل محلولة

٧ – ٥٠ أكتب شفرة كوبل لشكل ٧ – ١٩

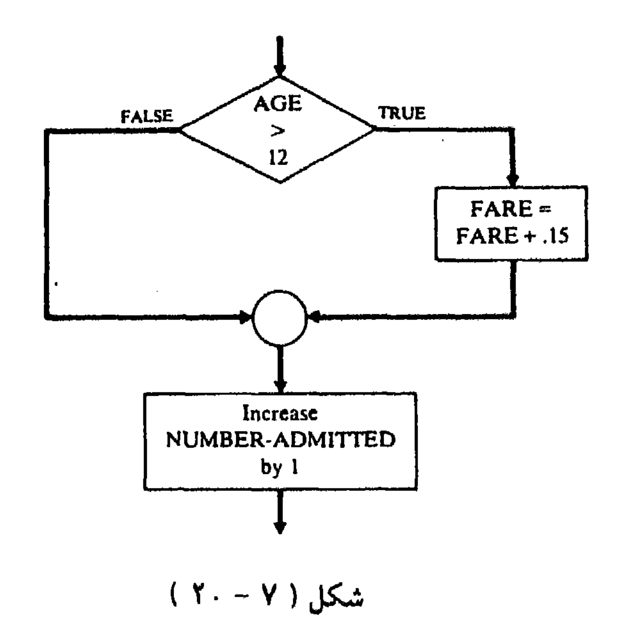
```
IF AGE IS GREATER THAN 12
    MOVE 4.00 TO FARE
ELSE
    MOVE 3.00 TO FARE

ADD 1 TO NUMBER-ADMITTED
```




شكل ( ٧- ١٩ )

٧ – ٥١ اكتب شفرة كوبل لشكل ( ٧ – ٢٠ )




شكل ( ٧ – ٢٠ )

يكتب معظم مبرمجى الكوبل ما يلى :

```
IF AGE GREATER THAN 12
    ADD .15 TO FARE

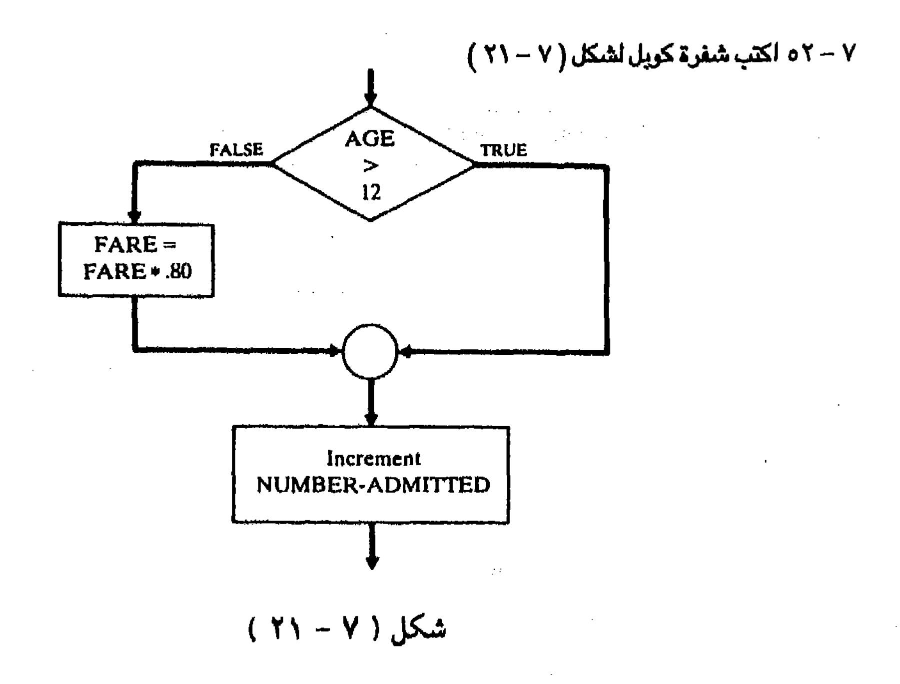
ADD 1 TO NUMBER-ADMITTED
```

إلا أن البعض يفضل وجود ELSE لكل IF :

```
IF AGE GREATER THAN 12
    ADD .15 TO FARE
ELSE
    NEXT SENTENCE

ADD 1 TO NUMBER-ADMITTED
```

٧ - ٥٢ اكتب شفرة كوبل لشكل ( ٧ - ٢١ )




شكل ( ٧ - ٢١ )

If NEXT SENTENCE is used, the solution is straightforward:

```
IF AGE GREATER THAN 12
    NEXT SENTENCE
ELSE
    COMPUTE FARE = FARE * .80

ADD 1 TO NUMBER-ADMITTED
```

قد يرى بعض المبرمجين أنه من الأفضل حذف NEXT SENTENCE بنفي الشرط فى شكل ٧ - ٢١ ، وتبادل TRUE مع FALSE . وهذا يعطى الشفرة المكافئة التالية :

```
IF AGE NOT GREATER THAN 12
    COMPUTE FARE = FARE * .80
.
ADD 1 TO NUMBER-ADMITTED
```

٧ – ٥٣ حدد الخطأ فيما يلي :

```
IF CUSTOMER-TYPE EQUAL "3"
    MOVE ZERO TO SHIPPING-DISCOUNT
ELSE
    PERFORM DETERMINE-SHIPPING-DISCOUNT
ADD SHIPPING-DISCOUNT TO TOTAL-DISCOUNT
PERFORM FORMAT-INVOICE
```

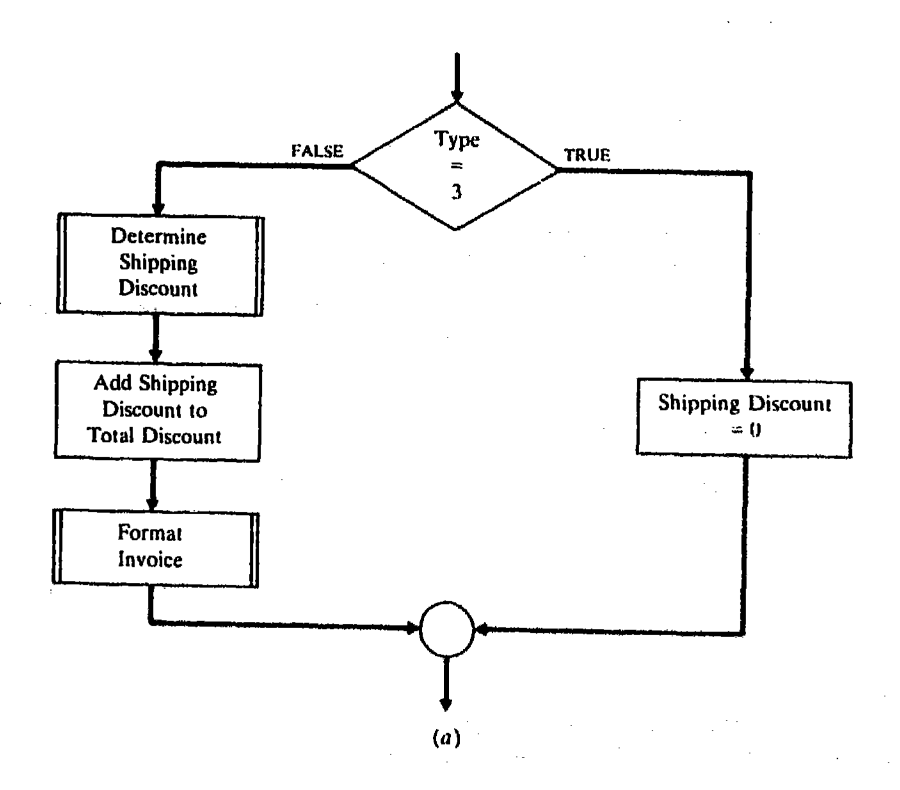
إذا حدد الترحيل التشغيل المطلوب بطريقة صحيحة .. فستكون هناك حاجة إلى نقطة لإنهاء عبارة IF كما يلي :

```
IF CUSTOMER-TYPE EQUAL "3"
    MOVE ZERO TO SHIPPING-DISCOUNT
ELSE
    PERFORM DETERMINE-SHIPPING-DISCOUNT

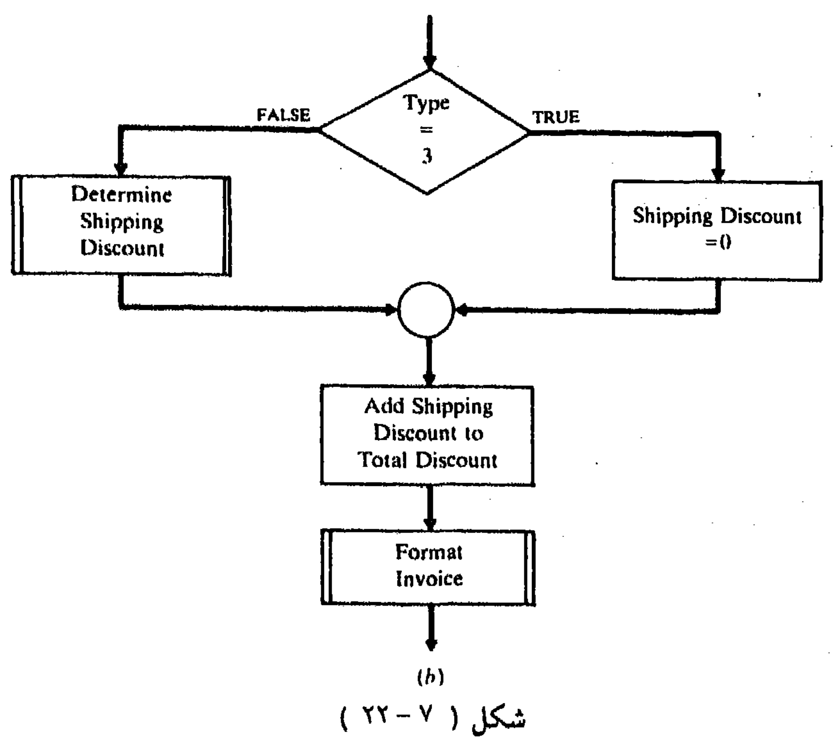
ADD SHIPPING-DISCOUNT TO TOTAL-DISCOUNT
PERFORM FORMAT-INVOICE
```

بدون النقطة .. تعتبر عبارة ADD وعبارة PERFORM جزءاً من ELSE ( انظر ملحق حـ بالنسبة إلى كوبل 1985 النمطى ) .

٧ – ٥٤ ارسم خريطة مسار لجزء برنامج التمرين السابق (أ) بدون نقطة (ب) وبالنقطة . انظر شكل ٧ – ٢٢ .




(a)




(*b*)

شكل ( ٧ - ٢٢ )

٧ - ٥٥ أعد كتابة عبارة IF التالية بدون استخدام NEXT SENTENCE .

```
IF NO-SPECIAL-DISCOUNTS
    NEXT SENTENCE
ELSE
     ADD 1 TO NUMBER-SPECIAL-CASES
     PERFORM DETERMINE-SPECIAL-DISCOUNT
```

```
IF NOT NO-SPECIAL-DISCOUNTS
    ADD 1 TO NUMBER-SPECIAL-CASES
    PERFORM DETERMINE-SPECIAL-DISCOUNT
```

وهناك حل أفضل ، وهو تعريف اسم شرطى جديد : SPECIAL - DISCOUNTS :

```
05  SPECIAL-DISCOUNT-SWITCH     PIC X.
    88  NO-SPECIAL-DISCOUNTS    VALUE "A".
    88  SPECIAL-DISCOUNTS       VALUE "B" THRU "G".
```

يمكن الآن حذف NOT كما يلى :

```
IF SPECIAL-DISCOUNTS
    ADD 1 TO NUMBER-SPECIAL-CASES
    PERFORM DETERMINE-SPECIAL-DISCOUNT
```

٧ – ٥٦ اكتب جزءا من برنامج كوبل لوضع YES فىINVALID - RECORD - SW ، اذا كان أحد الحقول التالية غير عددى : ACCOUNT - NUMBER و TRANSACTION - CODE و AMOUNT .

```
IF    ACCOUNT-NUMBER NOT NUMERIC
  OR TRANSACTION-CODE NOT NUMERIC
  OR AMOUNT NOT NUMERIC
          MOVE "YES" TO INVALID-RECORD-SW
```

٧ – ٥٧ صحح الشفرة التالية ( ارجع للتمرين السابق ) .

```
IF ACCOUNT-NUMBER NOT NUMERIC
    PERFORM INVALID-ACCOUNT
ELSE IF TRANSACTION-CODE NOT NUMERIC
    PERFORM INVALID-TRANSACTION
ELSE IF AMOUNT NOT NUMERIC
    PERFORM INVALID-AMOUNT
```

الخطأ هنا هو أن IF الخطية غير مناسبة لمثل هذا النوع من اختبار الصحة . بمجرد تحقق أحد الشروط .. يهمل بقية IF الخطية ، وهذا ليس مرغوبا فيه ؛ لأننا نريد الاستمرار فى اختبار الصحة بعد اكتشاف أى خطأ بحيث يمكن معرفة أى أخطاء أخرى فى السجل (وتصحيحها) .

والطريقة الصحيحة تستخدم مجموعة من عبارات IF المنفصلة على النحو التالى :

```
IF ACCOUNT-NUMBER NOT NUMERIC
    PERFORM INVALID-ACCOUNT
.
IF TRANSACTION-CODE NOT NUMERIC
    PERFORM INVALID-TRANSACTION
.
IF AMOUNT NOT NUMERIC
    PERFORM INVALID-AMOUNT
```

٧ – ٥٨ بمعرفة PICTURE والمحتويات التالية ، اذكر ما إذا كان عنصر البيانات عدديًا أم حرفيًا أم غير ذلك .

| | *PICTURE* | *Contents* |
|---|---|---|
| (*a*) | S99 | −82 |
| (*b*) | XXX | "E T" |
| (*c*) | S99 | 82 |
| (*d*) | 99 | 82 |
| (*e*) | 99 | +82 |
| (*f*) | XXX | "C3B" |
| (*g*) | XXX | "    " |

(a) عددى (b) حرفى (c) لا هذا ولا ذاك (الصورة بها إشارة ، ويجب أن تكون بالمحتويات إشارة ) . (d) عددى (e) لا هذا ولا ذاك ( عندما لا تكون هناك إشارة فى PILTURE فيجب ألا توجد إشارة فى المحتويات ) (f) لا هذا ولا ذاك (g) حرفى (الفراغات مسموح بها) .

٧ – ٥٩ اذكر ما اذا كان يجب التأكد من صحة عناصر البيانات التالية قبل التشغيل أم لا .

(أ) قيمة مبلغ مودع ببنك يقوم الصراف بإدخاله .

(ب) الاسم البريدى المشترك فى مجلة يتم إدخاله عن طريق إدخال البيانات .

(جـ) قيمة المشتروات على الحساب التى يدخلها موظف المحل البائع .

(د) موازنة الرصيد الدائن للعميل من (جـ) كما تقرأ من الملف الرئيسى للعملاء الموجود على قرص .

(هـ) موازنة الحساب السابق للحساب الموجود فى (أ) كما يقرأ من ملف العملاء الرئيسى ، الموجود على قرص .

(و) كمية العنصر المطلوبة من قبل العميل ، ويتم إدخالها فى مستند المصدر عن طريق إدخال البيانات .

(ز) سعر العنصر فى (و) الذى يقرأ من ملف مخزون رئيسى على قرص .

(أ) الحقل يكون مهما ، ويتم إدخاله بواسطة مشغل الكمبيوتر . وعلى هذا يجب التأكد من صحة .

(ب) هذا الحقل يدخله مشغل الكمبيوتر إلا أنه ليس مهما . يمكن ان تصل المجلة بصورة مناسبة إلى المقصد المقصود ، حتى إذا كان هناك خطأ فى الاسم . كما ان اختبار الفئة لبيانات حرفية يمكن أن يكتشف رموزاً غير عددية فقط وليس خطأ فى الهجاء . وعلى هذا .. فالتاكد من صحة هذا الحقل قد لا تستحق إجراؤها .

(جـ) يجب التأكد ستكون من صحته فى أول مرة يتم تشغيله ، حيث إنه يمثل بيانات حرجة ويدخله مشغل كمبيوتر .

(د) بافتراض أن كل البيانات موضوعة فى الملف الرئيسى سبق التأكد من صحتها .. فإن إعادة التأكد من صحتها ستكون مضيعة لوقت الكمبيوتر .

(هـ) نفس الشىء مثل (د) .

(و) يجب أن يتم التأكد من صحته فى أول مرة يجرى عليه تشغيل .

(ز) نفس الشىء مثل (د) .

٧ – ٦٠ بين العلاقات الجبرية التالية كما تبدو عند كتابتها بعبارة IF من الكوبل

$(a)\ a + b = 10 \qquad (b)\ 5a = \frac{b}{3} \qquad (c)\ a \neq b \qquad (d)\ \frac{a}{b} \leq c \qquad (e)\ a \geq b$

(*a*) IF A + B NOT GREATER THAN 10

(*b*) IF 5 * A NOT LESS THAN B / 3

(*c*) IF A NOT EQUAL TO B

(*d*) IF A / B NOT GREATER THAN C

(*e*) IF A NOT LESS THAN B

٧ – ٦١ حدد إذا كان العنصر A أقل من ، أو يساوى ، أو أكبر من العنصر B .

| | (a) | (b) | (c) | (d) | (e) | (f) | (g) |
|---|---|---|---|---|---|---|---|
| A | CAD | +72 | HI | −003 | −4 | +34.5 | +34.00 |
| B | COB | 0072 | HIGH | −3 | +4 | +34 | +34 |

(a) أقل من ( يحفظ الترتيب الأبجدى )

(b) يساوى ( مقارنة جبرية ) .

(c) أقل من (HI = HIbb)

(d) يساوى ( مقارنة جبرية )

(e) أقل من ( مقارنة جبرية )

(f) أكبر ( مقارنة جبرية )

(g) يساوى ( مقارنة جبرية )

٧ – ٦٢ أعد كتابة العبارات التالية : مستخدما شروط إشارة بدلا من شروط علاقات .

(*a*) IF AMOUNT-DUE IS GREATER THAN ZERO . . . ;
(*b*) PERFORM POST-GENERAL- LEDGER UNTIL LEDGER-ACCOUNT IS EQUAL TO ZERO;
(*c*) IF QUANTITY-ON- HAND IS LESS THAN ZERO . . . .

(*a*) IF AMOUNT-DUE IS POSITIVE . . . ; (*b*) PERFORM POST-GENERAL-LEDGER UNTIL LEDGER-ACCOUNT IS ZERO; (*c*) IF QUANTITY-ON-HAND NEGATIVE . . . .

لاحظ حذف IS من (C) .

٧ – ٦٣ أعد تعريفات البيانات التالية ، وعبارات IF و PERFORM ، مستخدما أسماء شرطية بدلا من شروط العلاقات :

(*a*)
```
05  CUSTOMER-ID     PIC X(5).

    IF CUSTOMER-ID NOT LESS THAN ZERO AND NOT GREATER
          THAN "11111"
       PERFORM PROCESS-PREFERRED-CUSTOMER
    .
```

(*b*)
```
05  TRANSACTION-CODE     PIC X.

    IF TRANSACTION-CODE EQUAL "1"
       PERFORM PROCESS-DEPOSIT
    ELSE IF TRANSACTION-CODE EQUAL "2"
       PERFORM PROCESS-WITHDRAWAL
```

```
      .
(c)    05  REGION-CODE     PIC X.

           IF REGION-CODE EQUAL "A" OR "B" OR "C"
               PERFORM SHIP-TO-NORTHEAST
           ELSE IF REGION-CODE GREATER THAN "C"
                 AND LESS THAN "L"
               PERFORM SHIP-TO-SOUTHEAST
           ELSE IF REGION-CODE EQUAL "M" OR "P"
               PERFORM SHIP-WEST
           .
(d)    05  YEAR-IN-SCHOOL     PIC 99.

           IF YEAR-IN-SCHOOL EQUAL 6
               PERFORM PROCESS-6TH-GRADER
           ELSE IF YEAR-IN-SCHOOL NOT LESS THAN 9 AND NOT
                 GREATER THAN 12
               PERFORM PROCESS-HIGH-SCHOOL

(e)    05  DEPRECIATION-TYPE     PIC 9.

           IF DEPRECIATION-TYPE EQUAL 1 OR 2 OR 3
               PERFORM USE-STRAIGHT-LINE-VALUE
           ELSE IF DEPRECIATION-TYPE EQUAL 4
               PERFORM USE-DOUBLE-DECLINING-VALUE

(a)    05  CUSTOMER-ID                PIC X(5).
           88  PREFERRED-CUSTOMER     VALUE ZERO THRU "11111".

           IF PREFERRED-CUSTOMER
               PERFORM PROCESS-PREFERRED-CUSTOMER

(b)    05  TRANSACTION-CODE           PIC X.
           88  DEPOSIT                VALUE "1".
           88  WITHDRAWAL             VALUE "2".

           IF DEPOSIT
               PERFORM PROCESS-DEPOSIT
           ELSE IF WITHDRAWAL
               PERFORM PROCESS-WITHDRAWAL

(c)    05  REGION-CODE                PIC X.
           88  NORTHEAST-REGION       VALUE "A" THRU "C".
           88  SOUTHEAST-REGION       VALUE "D" THRU "K".
           88  WESTERN-REGION         VALUE "M"  "P".

           IF NORTHEAST-REGION
               PERFORM SHIP-TO-NORTHEAST
           ELSE IF SOUTHEAST-REGION
               PERFORM SHIP-TO-SOUTHEAST
           ELSE IF WESTERN-REGION
               PERFORM SHIP-WEST
(d)    05  YEAR-IN-SCHOOL             PIC 99.
           88  6TH-GRADE              VALUE 6.
           88  HIGH-SCHOOL            VALUE 9 THRU 12.
```

```
IF 6TH-GRADE
    PERFORM PROCESS-6TH-GRADER
ELSE IF HIGH-SCHOOL
    PERFORM PROCESS-HIGH-SCHOOL.
```

(*e*)

```
05  DEPRECIATION-TYPE          PIC 9.
    88  STRAIGHT-LINE          VALUE 1 THRU 3.
    88  DOUBLE-DECLINING       VALUE 4.

    IF STRAIGHT-LINE
        PERFORM USE-STRAIGHT-LINE-VALUE
    ELSE IF DOUBLE-DECLINING
        PERFORM USE-DOUBLE-DECLINING-VALUE
```

٧ – ٦٤ صحح العبارات التالية ، والتى تستخدم الأسماء الشرطية من التمرين رقم ٦٣ استخداما خاطئا .

(*a*) IF PREFERRED-CUSTOMER EQUAL "00111"

(*b*) IF DEPOSIT
MOVE SPACES TO WITHDRAWAL

(*c*) IF HIGH-SCHOOL EQUAL 10

(أ) IF CUSTOMER-ID EQUAL "00111"

لا يمكن استخدام اسم شرطى ، بدلا من اسم بيانات بأى حال من الأحوال فى البرنامج . الاسم الشرطى هو اختصار فقط لشرط علاقة محدد .

(ب)

```
IF DEPOSIT
    MOVE SPACES TO TRANSACTION-CODE
```

كما فى (أ) .. يجب استخدام عنصر البيانات فى عبارة MOVE ، وليس اسماً شرطياً .

(جـ) IF YEAR-IN-SCHOOL EQUAL 10

لا يوجد اسم شرطى للعلاقة :

YEAR - IN - SCHOOL EQUAL 10 ، وعلى هذا .. يجب كتابة الشرط فى عبارة IF . والبديل لذلك هو عمل شرط اسم شرطى كما يلى :

```
05  YEAR-IN-SCHOOL      PIC 99.
    88  10TH-GRADE      VALUE 10.

    IF 10TH-GRADE
```

٧ – ٦٥ هناك قاعدة فى الكوبل تنص على أن جزء VALUE لا يمكن استخدامه فى قسم الملفات . هل ينطبق ذلك على الأسماء الشرطية؟

لا ، تنطبق القاعدة على عناصر البيانات فى قسم الملفات فقط . يمكن (ويجب) استخدام جزء VALUE فى قسم الملفات عند تعريف أسماء شرطية (عناصر على المستوى 88 ) .

٧ – ٦٦ إذا كانت C, B, A أسماء شرطية .. افرض أن B, A صحيحان بينما C خطأ . حدد ما إذا كانت الشروط المركبة التاليه صحيحة أم خطأ .

(*a*) A AND B OR A AND C
(*b*) A AND (B OR A) AND C
(*c*) A AND NOT C OR D
(*d*) A AND NOT (C OR D)
(*e*) A AND NOT C OR NOT D
(*f*) (A AND NOT C) OR NOT D
(*g*) A AND NOT (C OR NOT D)
(*h*) A AND C OR B AND D
(*i*) A AND B AND C
(*j*) A OR B OR C OR D
(*k*) A OR C AND B OR D
(*l*) (A OR C) AND (B OR D)
(*m*) A AND C OR B OR D

(a) فى غياب الأقواس .. تنفذ AND أولا (من اليسار إلى اليمين) يتبعها تنفيذ OR . وعلى هذا .. يتم تقويم التعبير كما يلى: (A AND B) OR (A AND C) . (A AND B) صحيح وبالتالى فان .. (A AND B) OR يكون صحيحا كذلك .

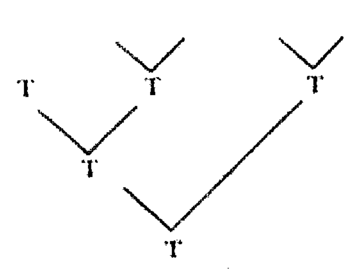
(b) يتم تقويم التعبير الموجود بين قوسين أولا : (B OR A) صحيح . ثم ينفذ AND من اليسار لليمين : A AND (B OR A) يكون صحيحا ، هذه النتيجة مرتبطة بواسطة AND مع C وينتج عنها نتيجة خطأ ، وهى النتيجة النهائية .

(c) يتم تقويم NOT أولاً لتعطى النتيجة صحيح، ويلى ذلك تقويم A AND NOT ، وتكون صحيحة. D مع هذه النتيجة بواسطة OR ، وبالتالى فالنتيجة النهائية صحيحة.

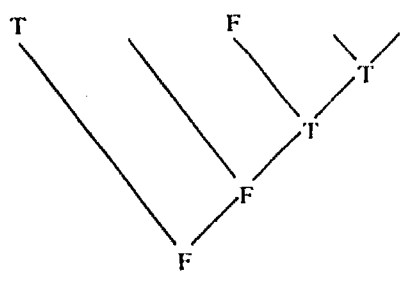
(d) (C OR D) تعطى نتيجة خطأ وبالتالى NOT (C OR D) تعطى نتيجة صحيحة . وهذه مرتبطه مع A بواسطة AND فتعطى النتيجة النهائية صحيحة .

(e) تنفذ NOT أولا من اليسار لليمين نتيجة NOT C هى صحيحه، كما أن نتيجة NOT D صحيحه. ترتبط A مع NOT C باستخدام AND وتعطى النتيجة صحيحة . ترتبط هذه النتيجة بواسطة OR مع NOT D لتكون النتيجة النهائية صحيحة.

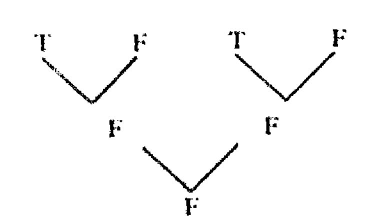
(*f*) (A AND NOT C) OR NOT D



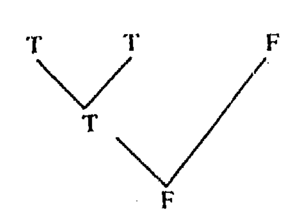

(*g*) A AND NOT (C OR NOT D)



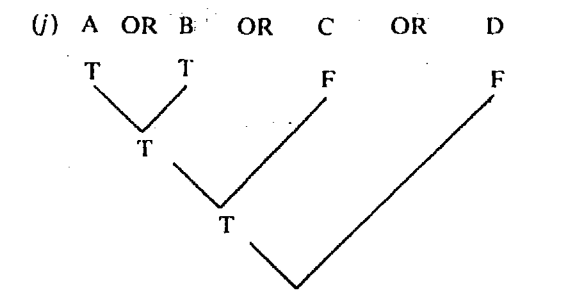

(*h*) A AND C OR B AND D



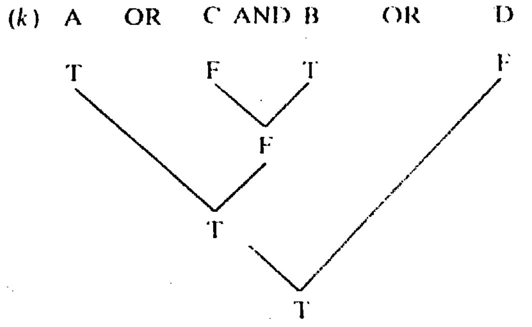

(*i*) A AND B AND C



(*j*) A OR B OR C OR D



(*k*) A OR C AND B OR D



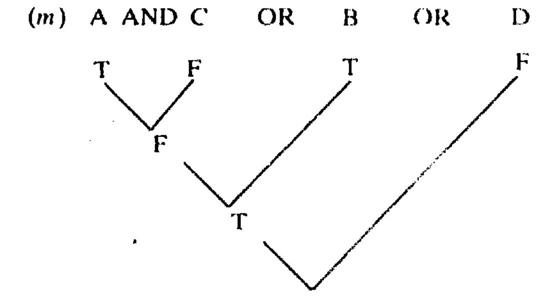
(*l*) (A OR C) AND (B OR D)

T F T F

T T

T

(*m*) A AND C OR B OR D



٧ – ٦٧ افترض أن D, C, B, A هي أسماء شرطية . اعد كتابة الشروط المركبة التالية مستخدما أقواسًا ؛ لتوضيح ترتيب تنفيذ العمليات .

(*a*) NOT A OR B AND NOT C
(*b*) NOT A OR NOT B AND NOT C
(*c*) A AND C OR B
(*d*) A OR B AND C

(*a*) (NOT A) OR (B AND (NOT C))
(*b*) (NOT A) OR ((NOT B) AND (NOT C))
(*c*) (A AND C) OR B
(*d*) A OR (B AND C)

٧ – ٦٨ اكتب شرطا يكون صحيحا إذا ، واذا فقط ، وقع ITEM-A بين 27,3 شاملا الحدود . ابسط حل هو :

IF ITEM-A NOT LESS THAN 3 AND NOT GREATER THAN 27

٧ – ٦٩ اكتب شرطا يكون صحيحا إذا ، واذا فقط ، مالم يقع ITEM-A بين 27,3 شاملا الحدود .

IF ITEM-A LESS THAN 3 OR GREATER THAN 27

لاحظ استخدام OR بدلا من AND : أحد الشروط البسيطة فقط يجب أ ن يكون صحيحا ليقع ITEM-A خارج المدى المحدد .

٧ - ٧٠ اكتب شرطا يكون صحيحا إذا ، وإذا فقط ، كان أى من - PAYMENT - DUE , PAYMENT , OLD BALANCE غير عددى .

```
IF     OLD-BALANCE NOT NUMERIC
   OR PAYMENT NOT NUMERIC
   OR PAYMENT-DATE NOT NUMERIC
```

٧ - ٧١ اكتب شرطا يكون صحيحا إذا ، وإذا فقط ، كان A مساويا 1 ، و B مساويا 2 ، و D ليس مساويا 4 .

```
IF       A EQUAL 1
   AND B EQUAL 2
   AND C EQUAL 3
   AND D NOT EQUAL 4
```

٧ - ٧٢ اكتب شرطا يكون صحيحا إذا ، وإذا فقط ، لم يكن END-OF-FILE صحيحا ، وإما أن يكون BALANCE سالبا أو يكون PAYMENT مساويا صفر .

```
IF       NOT END-OF-FILE
   AND (BALANCE IS NEGATIVE OR PAYMENT IS ZERO)
```

الأقواس مطلوبة : حيث يمكن تقويم A AND B OR C على أنها (A AND B) OR C .

٧ - ٧٣ اكتب شرطا يكون صحيحا إذا ، وإذا فقط ، كانت YEAR تقع بين 85,82 ، وتقع MONTH بين 8,6 ، وتقع DAY بين 20,5 .

```
IF       (YEAR NOT LESS THAN 82 AND NOT GREATER THAN 85)
   AND (MONTH NOT LESS THAN 6 AND NOT GREATER THAN 8)
   AND (DAY NOT LESS THAN 5 AND NOT GREATER THAN 20)
```

حيث إن المؤثر المنطقى AND هو المطلوب فقط فكل الأقواس اختيارية (بالرغم من أنها تحسن من التوضيح) .

٧ - ٧٤ بين كيف يمكن اختصار شروط العلاقات التالية :

(a) A NOT EQUAL 1 AND A NOT EQUAL 2 AND A NOT EQUAL 3
(b) A EQUAL 7 OR A EQUAL 8 AND B EQUAL 8
(c) A EQUAL 1 OR A EQUAL 2 AND A EQUAL 3
(d) A EQUAL 1 OR A EQUAL 2 OR A LESS THAN 9 OR B LESS THAN 10
(e) A EQUAL 1 OR A LESS THAN 5 OR A GREATER THAN 8
(f) A EQUAL B AND A LESS THAN C AND A LESS THAN D

(a) A NOT EQUAL 1 AND 2 AND 3
(b) A EQUAL 7 OR 8 AND B EQUAL 8
(c) A EQUAL 1 OR 2 AND 3 (which can be logically reduced to A EQUAL 1)
(d) A EQUAL 1 OR 2 OR LESS THAN 9 OR B LESS THAN 10
(e) A EQUAL 1 OR LESS THAN 5 OR GREATER THAN 8
(f) A EQUAL B AND LESS THAN C AND D

٧ - ٧٥ اكتب شروط العلاقات التالية فى صوره غير مختصرة .

(a) A EQUAL B OR C AND LESS THAN D
(b) A GREATER THAN B OR D AND C OR LESS THAN B OR D
(c) A LESS THAN B AND D AND C
(d) A EQUAL B OR D AND C
(e) A EQUAL B OR C AND LESS THAN D OR GREATER THAN E
(f) A NOT EQUAL B OR D AND NOT LESS THAN C AND E

(a) A EQUAL B OR A EQUAL C AND A LESS THAN D
(b) A GREATER THAN B OR A GREATER THAN D AND A GREATER THAN C OR A LESS THAN B OR A LESS THAN D
(c) A LESS THAN B AND A LESS THAN D AND A LESS THAN C
(d) A EQUAL B OR A EQUAL D AND A EQUAL C
(e) A EQUAL B OR A EQUAL C AND A LESS THAN D OR A GREATER THAN E
(f) A NOT EQUAL B OR A NOT EQUAL D AND A NOT LESS THAN C AND A NOT LESS THAN E

٧ - ٧٦ راجع الترحيل المضلل التالى لتبين كيف تنفذ IF المتداخلة فعلا .

(a)
```
IF NOT END-OF-FILE
    IF CREDIT-LINE-OK
        PERFORM DETERMINE-CREDIT-LIMIT
        PERFORM PRINT-TERMS
ELSE
    PERFORM PRINT-FINAL-TOTALS
```

(*b*)

```
IF PAYMENT NOT NUMERIC
    PERFORM INVALID-PAYMENT
    IF DATE2 NOT NUMERIC
        PERFORM INVALID-DATE2
ELSE
    PERFORM APPLY-PAYMENT
    IF BALANCE NEGATIVE
        PERFORM GIVE-REFUND
ELSE
    PERFORM FILE-BALANCE
```

(*c*)

```
IF A EQUAL B
MOVE 1 TO A
MOVE 2 TO B.
MOVE 3 TO C

IF B EQUAL C
    MOVE 4 TO D
    IF C EQUAL D
    MOVE 5 TO E
ELSE
    MOVE 6 TO F.
```

(*a*)

```
IF NOT END-OF-FILE
    IF CREDIT-LINE-OK
        PERFORM DETERMINE-CREDIT-LIMIT
        PERFORM PRINT-TERMS
    ELSE
        PERFORM PRINT-FINAL-TOTALS
```

(*b*)

```
IF PAYMENT NOT NUMERIC
    PERFORM INVALID-PAYMENT
    IF DATE2 NOT NUMERIC
        PERFORM INVALID-DATE2
    ELSE
        PERFORM APPLY-PAYMENT
        IF BALANCE NEGATIVE
            PERFORM GIVE-REFUND
        ELSE
            PERFORM FILE-BALANCE
```

هنا ، كما فى (أ) ، كل جزء ELSE متزامل مع أول IF تسبقه .

(*c*)

```
IF A EQUAL B
    MOVE 1 TO A
    MOVE 2 TO B

MOVE 3 TO C
IF B EQUAL C
    MOVE 4 TO D
    IF C EQUAL D
        MOVE 5 TO E
```

```
ELSE
    MOVE 6 TO F
```

لاحظ أهمية النقطة فى تغيير معنى IF المتداخلة . بدون النقطة بعد MOVE 2 TO B .. ينفذ التسلسل كما يلى :

```
IF A EQUAL B
    MOVE 1 TO A
    MOVE 2 TO B
    MOVE 3 TO C
    IF B EQUAL C
        MOVE 4 TO D
        IF C EQUAL D
            MOVE 5 TO E
        ELSE
            MOVE 6 TO F
```

٧ – ٧٧ اعد كتابة ما يلى دون استخدام NEXT SENTENCE ، صحح كذلك الترحيل :

```
IF A EQUAL B
    NEXT SENTENCE
ELSE
    PERFORM PARA-1
    PERFORM PARA-2.
    PERFORM PARA-3

IF A NOT EQUAL B
    PERFORM PARA-1
    PERFORM PARA-2

PERFORM PARA-3
```

٧ – ٧٨ أكتب شفرة هيكل الحالة التالى فى مقطع مستخدما IF الخطية .

| ***REGION-CODE Value*** | ***Action*** |
|---|---|
| 1 | PERFORM SHIP-NORTHEAST |
| 2 | PERFORM SHIP-SOUTHEAST |
| 3 | PERFORM SHIP-NORTHEAST |
| 4 | PERFORM SHIP-WEST |

```
DETERMINE-SHIPPING.
    IF REGION-CODE EQUAL 1
        PERFORM SHIP-NORTHEAST
    ELSE IF REGION-CODE EQUAL 2
        PERFORM SHIP-SOUTHEAST
    ELSE IF REGION-CODE EQUAL 3
        PERFORM SHIP-NORTHEAST
    ELSE IF REGION-CODE EQUAL 4
        PERFORM SHIP-WEST
    ELSE
        PERFORM INVALID-CODE
```

أو بدمج الحالات :

```
DETERMINE-SHIPPING.
    IF REGION-CODE EQUAL 1 OR 3
        PERFORM SHIP-NORTHEAST
    ELSE IF...
```

٧ – ٧٩ وضح كيف ينفذ هيكل الحالة فى التمرين السابق ( ٧ – ٧٨ ) .

```
PERFORM DETERMINE-SHIPPING
```

٧ – ٨٠ اعد كتابة ما يلى مستخدما PERFORM ... UNTIL .

```
PERFORM GET-INPUT
PERFORM GET-INPUT
PERFORM GET-INPUT
PERFORM GET-INPUT
PERFORM GET-INPUT
    VARYING TIMES-COUNTER
        FROM 1 BY 1
    UNTIL TIMES-COUNTER GREATER THAN 4
```

٧ – ٨١ كم عدد المرات التى ينفذ فيها GET-INPUT ؟

```
PERFORM GET-INPUT
    VARYING TIMES-COUNTER FROM 1 BY 1
    UNTIL TIMES-COUNTER EQUAL 10
```

٩ مرات .

٧ – ٨٢ كم عدد المرات التى ينفذ فيها PARA-A ؟

```
    MOVE 10 TO TIMES-COUNTER
    PERFORM PARA-A TIMES-COUNTER TIMES
    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
PARA-A.
    MOVE 5 TO TIMES-COUNTER
    MOVE P TO Q
```

٧ – ٨٣ كم عدد المرات التى ينفذ فيها PARA - A ؟

```
PERFORM PARA-A
    VARYING TIMES-COUNTER FROM 1 BY 1
    UNTIL TIMES-COUNTER GREATER THAN 10
    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    PARA-A.
        MOVE P TO Q
        MOVE 11 TO TIMES-COUNTER
```

ينفذ PARA-A مرة واحدة . عندما تستخدم PERFORM ... VARYING .. UNTIL ، فأى تغيير فى عنصر بيانات VARYING ، أو فى قيمة BY أو فى قيمة UNTIL يؤثر على عدد مرات التكرار .

٧ – ٨٤ كم عدد المرات التى ينفذ فيها PARA-A ؟

```
MOVE 11 TO TIMES-COUNTER
PERFORM PARA-A
    UNTIL TIMES-COUNTER GREATER THAN 10
```

لا ينفذ PARA-A على الإطلاق ، ويتم تقويم شرط UNTIL قبل أى PERFORM .

٧ – ٨٥ كم عدد مرات التى ينفذ فيها PARA-S ؟

```
001      MOVE ZERO TO TIMES-COUNTER
         PERFORM PARA-S
             UNTIL TIMES-COUNTER GREATER THAN 10
         . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
         PARA-S.
             ADD 1 TO TIMES-COUNTER
             DISPLAY TIMES-COUNTER
```

١١ مرة: TIMES - COUNTER يعرض القيم 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11 .

٧ – ٨٦ حدد قيم ITEM لكل تنفيذ للمقطع PARA-A :

```
(a)    PERFORM PARA-A VARYING ITEM FROM 1 BY 2
           UNTIL ITEM GREATER THAN 8
(b)    PERFORM PARA-A
           VARYING ITEM FROM 10 BY -3
           UNTIL ITEM LESS THAN -4
(c)    PERFORM PARA-A
           VARYING ITEM FROM 2.7 BY -.4
           UNTIL ITEM NEGATIVE
(d)    PERFORM PARA-A
           VARYING ITEM FROM 6 BY 2
           UNTIL ITEM EQUAL 11
(e)    PERFORM PARA-A
           VARYING ITEM FROM 5 BY 5
           UNTIL ITEM GREATER THAN 25
```

(a) 7, 5, 3, 1 يتوقف التنفيذ عند ITEM = 9 .

(b) 10, 7, 4, 1, 2 – يتوقف التنفيذ عند ITEM = - 5 .

(c) 2.7, 2.3, 1.9, 1.5, 1.1, 0.7, 0.3 ويتوقف التنفيذ عند ITEM = - 0.1 .

(d) 6, 8, 10, 12, 14, 16 .... حتى ينهى نظام التشغيل الوقت ، أو يزيد عدد السجلات عن العدد المحدد من قبل نظام التشغيل . هذه دورة لا نهائية .

(e) 5, 10, 15, 20, 50 ، ويتوقف التنفيذ عند ITEM = 30 .

٧ – ٨٧ حدد قيم B,A التى ينفذ عندها PARA .

```
PERFORM PARA
    VARYING A FROM 2 BY 2
        UNTIL A GREATER THAN 8
    AFTER B FROM 5.5 BY -.5
        UNTIL B LESS THAN 4.0
```

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 2 | 2 | 2 | 2 | 4 | 4 | 4 | 4 | 6 | 6 | 6 | 6 | 8 | 8 | 8 | 8 |
| B | 5.5 | 5.0 | 4.5 | 4.0 | 5.5 | 5.0 | 4.5 | 4.0 | 5.5 | 5.0 | 4.5 | 4.0 | 5.5 | 5.0 | 4.5 | 4.0 |

ينتهى تنفيذ عبارة PERFORM عند A=10, B = 5.5 .

٧ – ٨٨ حدد قيم C,B,A التى ينفذ عندها PARA .

```
PERFORM PARA
    VARYING A FROM 2 BY 3
        UNTIL A GREATER THAN 7
    AFTER B FROM 5 BY -1
        UNTIL B LESS THAN 3
    AFTER C FROM 1 BY 1
        UNTIL C GREATER THAN 3
```

| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| B | 5 | 5 | 5 | 4 | 4 | 4 | 3 | 3 | 3 | 5 | 5 | 5 | 4 | 4 | 4 | 3 | 3 | 3 |
| C | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |

ينتهى تنفيذ عبارة PERFORM عند C= 1, A= 8 ، و B = 5 .

٧ – ٨٩ وضح الخطأ بالنسبة للأجزاء التالية :

(a)
```
PERFORM PARA-A
    VARYING ITEM-A FROM 1 BY 1 UNTIL ITEM-A GREATER THAN 10
    VARYING ITEM-B FROM 5 BY 2 UNTIL ITEM-B GREATER THAN 20
```

(b)
```
    PERFORM PARA-A THRU PARA-C
    . . . . . . . . . . . . . . . . . . . . .

PARA-C.
    . . . . . . . . . . . . . . . . . . . . .
```

```
        PARA-A.
            . . . . . . . . . . . . . . . . . . . . . .
(c)         PERFORM PRODUCE-REPORT
                UNTIL END-OF-FILE
            . . . . . . . . . . . . . . . . . . . . . . . .
        PRODUCE-REPORT
            IF INPUT-FIELD NUMERIC
                PERFORM PROCESS-IT
            ELSE
                PERFORM INVALID-FIELD

            . . . . . . . . . . . . . . . . . . . . . . . .
        INVALID-FIELD.
            PERFORM WRITE-ERROR-MESSAGE
            PERFORM READ-NEXT-RECORD
            PERFORM PRODUCE-REPORT
```

(a) خطأ تكوينى : PERFORM مع عناصر متغيرة عديدة يجب أن تكتب فى صورة ... PERFORM ... VARYING ... AFTER ... AFTER ... .

(b) خطأ تكوينى : يجب على مقطع (أو قسم) THRU أن يتبع واقعيا أول مقطع (أو قسم) فى المدى .

(c) خطأ منطقى : PERFORM لها إعادة ذاتية . تنفذ عبارة PERFORM الأولية المقطع PRODUCE - REPORT . طبقا لصحة المدخلات ، يمكن للمقطع PRODUCE - REPORT أن ينفذ PERFORM المقطع INVALID - FIELD . عند ذلك .. ينفذ المقطع INVALID - FIELD المقطع PRODUCE - REPORT . وعلى هذا .. فمن الممكن تنفيذ PER- FORM INVALID - FIELD فى PRODUCE - REPORT قبل انتهاء أول تنفيذ لها . تنفيذ عبارة PERFORM ، التى لا تكون قد انتهى تنفيذها السابق (لان التحكم لم يعد من المقطع المنفذ) عادة ما تنتج عن دورة نهائية .

٧ – ٩٠ أكتب شفرة شبيهة لبرنامج لموازنة دفتر شيكات . يدخل الخوارزمى الموازنة القديمة تتبعها مجموعة من رموز العمليات الجارية (D للإيداع ، و C لقيمة الشيك المسحوب) والمبالغ . حتى نهاية الملف .. المخرجات هى إجمالى الإيداعات وإجمالى الشيكات والموازنة الجديدة .

```
read old-balance
set total-deposits, total-checks to zero
set end-sw to "no"
read transaction-code, amount
    at end set end-sw to "yes"
perform until end-sw = "yes"
    if transaction-code = "D"
        add amount to total-deposits
    else
        if transaction-code = "C"
            add amount to total-checks
        else
            print "invalid transaction code"
        endif
    endif
    read transaction-code, amount
        at end set end-sw = "yes"
endperform
new-balance = old-balance + total-deposits - total-checks
print old-balance, total-deposits, total-checks, new-balance
stop
```

٧ - ٩١ اكتب شفرة شبيهة لتنفيذ جدول القرارات الموجود فى شكل ٧ - ١٨

```
if account not active
    perform disapprove-purchase
else
    if last month's payment received
        if purchase not greater than credit limit
            perform approve-purchase
        else
            perform approve-purchase
            perform refer-to-credit-office
        endif
    else
        if purchase not greater than credit limit
            perform approve-purchase
            perform refer-to-credit-office
        else
            perform disapprove-purchase
        endif
    endif
endif
```

بينما يكون هذا الحل مباشرا ، إلا أن هناك حلاً أكثر إيجازا ، وهو ما يلى :

```
if account not active
    perform disapprove-purchase
else
    if last payment received and purchase within credit limit
        perform approve-purchase
    else
        if last payment not received and purchase not within credit limit
            perform disapprove-purchase
        else
            perform approve-purchase
            perform refer-to-credit-office
        endif
    endif
endif
```

٧ - ٩٢ بمعرفة الشروط التالية والإجراءات .. ارسم جدول قرارات لقبول عمل فى تشغيل البيانات .

| الشروط | الإجراءات |
|---|---|
| (١) الراتب والمزايا مقبولة | (١) اقبل العمل |
| (٢) تعليم وتدريب أكثر متاح | (٢) أرفض العمل |
| (٣) فرص تقدم جيدة | (٣) ناقش للوصول الى مزايا أفضل |

انظر شكل ٧ - ٢٣ لرؤية حل ممكن .

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Salary | T | T | T | T | F | F | F | F |
| Education | T | T | F | F | T | T | F | F |
| Advancement | T | F | T | F | T | F | T | F |
| Accept | × | × | × | × | × | | | |
| Refuse | | | | | | × | × | × |
| Negotiate | | | | × | × | × | × | |

شكل ( ٧ - ٢٣ )

## مشاكل متكاملة

٧ - ٩٣ اكتب برنامجا يمكن استخدامه فى أحد البنوك فى طباعة جداول دفع حصص ديون العملاء . مدخلات البرنامج هى قيمة أصل الدين ، ومعدل الفائدة السنوية ( كنسبة مئوية ) والقسط الشهرى المرغوب فى دفعه . يطبع البرنامج تقريرا يسرد المعلومات التالية لكل شهر لفترة إعادة دفع الدين :

(١) رقم القسط .

(٢) الموازنة القديمة ( قبل دفع هذا القسط) .

(٣) قيمة القسط .

(٤) قيمة ما دفع من أصل الدين [(٢) - (٥)] .

(٥) قيمه ما دفع من الفوائد [ (٢) x $\frac{\text{معدل الفائد}}{12}$ ]

(٦) الموازنة الجديدة [(٢) - (٤)] . يجب أن يطبع البرنامج 20 سطرا ، بينها مسافات مزدوجة فى الصفحة الواحدة . استخدم عناوين أعمدة مناسبة وأرقام صفحات ، وتاريخ التنفيذ وما إلى ذلك من معلومات . انظر شكل ٧ - ٢٤ للحل ، وشكل (٧-٢٥) لعينة مخرجات .

```
IDENTIFICATION DIVISION.

PROGRAM-ID.  MORTGAGE.

AUTHOR. LARRY NEWCOMER.
INSTALLATION. PENN STATE UNIVERSITY -- YORK CAMPUS.
DATE-WRITTEN.  MAY  1983.
SECURITY.  NONE.

ENVIRONMENT DIVISION.

CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-3081.
OBJECT-COMPUTER.  IBM-3081.

INPUT-OUTPUT SECTION.
FILE-CONTROL.
```

```

     SELECT TERMS-FILE
         ASSIGN TO TERMS
         ORGANIZATION IS SEQUENTIAL
         ACCESS IS SEQUENTIAL
         .
     SELECT PAYMENT-REPORT
         ASSIGN TO PAYMENTS
         ORGANIZATION IS SEQUENTIAL
         ACCESS IS SEQUENTIAL
         .

 DATA DIVISION.

 FILE SECTION.

 FD  TERMS-FILE
     RECORD CONTAINS 80 CHARACTERS
     LABEL RECORDS ARE OMITTED
     .

 01  TERMS-RECORD.
     05  TERMS-PRINCIPAL          PIC S9(7)V99.
     05  TERMS-INTEREST-RATE      PIC SV999.
     05  TERMS-PAYMENT            PIC S9(5)V99.
     05  FILLER                   PIC X(61).

 FD  PAYMENT-REPORT
     RECORD CONTAINS 132 CHARACTERS
     LABEL RECORDS ARE OMITTED
         LINAGE IS 60
             WITH FOOTING AT 59
             LINES AT TOP 3
             LINES AT BOTTOM 3


 01  PAYMENT-RECORD               PIC X(132).

 WORKING-STORAGE SECTION.

 01  WS-SWITCHES-AND-COUNTERS.
     05  WS-PAGE-NUMBER           PIC S999         COMP-3.
     05  WS-TERMS-FILE-END-SW     PIC XXX.
         88  EMPTY-TERMS-FILE     VALUE "YES".
     05  WS-FINAL-PAYMENT-SW      PIC XXX.
         88  FINAL-PAYMENT        VALUE "YES".
         88  REGULAR-PAYMENT      VALUE "NO".

 01  WS-COMPUTATION-WORK-AREAS.
     05  WS-INTEREST-AMOUNT       PIC S9(5)V99     COMP-3.
     05  WS-PAYMENT-AMOUNT        PIC S9(5)V99     COMP-3.
     05  WS-BALANCE               PIC S9(7)V99     COMP-3.
     05  WS-PAYMENT-NUMBER        PIC S9(3)        COMP-3.
     05  WS-INTEREST-RATE         PIC SV999        COMP-3.
     05  WS-PRINCIPAL-AMOUNT      PIC S9(5)V99     COMP-3.

 01  WS-DATE.
     05  SYSTEM-YY                PIC 99.
     05  SYSTEM-MM                PIC 99.
     05  SYSTEM-DD                PIC 99.

 01  WS-HEADING-LINE-1.
     05  FILLER                   PIC X(10)        VALUE SPACES.
     05  FILLER                   PIC X(17)
                                  VALUE "MORTGAGE PAYMENTS".
     05  FILLER                   PIC X(1)         VALUE SPACES.
     05  WS-HEADING-MM            PIC Z9.
```

```
           05  FILLER                      PIC X          VALUE "/".
           05  WS-HEADING-DD               PIC 99.
           05  FILLER                      PIC X          VALUE "/".
           05  WS-HEADING-YY               PIC 99.
           05  FILLER                      PIC X(4)       VALUE SPACES.
           05  FILLER                      PIC X(5)       VALUE "PAGE ".
           05  WS-HEADING-PAGE-NO          PIC ZZ9.
           05  FILLER                      PIC X(84)      VALUE SPACES.

       01  WS-HEADING-LINE-2.
           05  FILLER                      PIC X(10)
                                           VALUE "PAYMENT #".
           05  FILLER                      PIC X(2)       VALUE SPACES.
           05  FILLER                      PIC X(15)
                                           VALUE "OLD BALANCE".
           05  FILLER                      PIC X(2)       VALUE SPACES.
           05  FILLER                      PIC X(10)
                                           VALUE "PAYMENT".
           05  FILLER                      PIC X(2)       VALUE SPACES.
           05  FILLER                      PIC X(10)
                                           VALUE "PRINCIPAL".
           05  FILLER                      PIC X(2)       VALUE SPACES.
           05  FILLER                      PIC X(10)
                                           VALUE "INTEREST".
           05  FILLER                      PIC X(2)       VALUE SPACES.
           05  FILLER                      PIC X(12)
                                           VALUE "NEW BALANCE".
           05  FILLER                      PIC X(55)      VALUE SPACES.

       01  WS-DETAIL-LINE.
           05  FILLER                      PIC X(3)       VALUE SPACES.
           05  WS-DETAIL-NUMBER            PIC Z9.
           05  FILLER                      PIC X(7)       VALUE SPACES.
           05  WS-DETAIL-OLD-BAL           PIC Z,ZZZ,ZZZ.99.
           05  FILLER                      PIC X(5)       VALUE SPACES.
           05  WS-DETAIL-PAYMENT           PIC ZZ,ZZZ.99.
           05  FILLER                      PIC X(3)       VALUE SPACES.
           05  WS-DETAIL-PRINCIPAL         PIC ZZ,ZZZ.99.
           05  FILLER                      PIC X(3)       VALUE SPACES.
           05  WS-DETAIL-INTEREST          PIC ZZ,ZZZ.99.
           05  FILLER                      PIC X(3)       VALUE SPACES.
           05  WS-DETAIL-NEW-BAL           PIC Z,ZZZ,ZZZ.99.
           05  FILLER                      PIC X(55)      VALUE SPACES.

       PROCEDURE DIVISION.

       000-EXECUTIVE-MODULE.

           PERFORM 100-GET-TERMS
           PERFORM 200-PRODUCE-DETAIL-LINE
               UNTIL FINAL-PAYMENT
           PERFORM 300-PRODUCE-FINAL-PAYMENT
           PERFORM 400-TERMINATE
           STOP RUN
           .

       100-GET-TERMS.

           MOVE "NO" TO WS-TERMS-FILE-END-SW
                        WS-FINAL-PAYMENT-SW
           MOVE ZERO TO WS-PAGE-NUMBER
                        WS-PAYMENT-NUMBER
           OPEN    INPUT   TERMS-FILE
                   OUTPUT  PAYMENT-REPORT
           READ TERMS-FILE
```

```
          AT END
              MOVE "YES" TO WS-TERMS-FILE-END-SW
      .
      IF EMPTY-TERMS-FILE
          MOVE "YES" TO WS-FINAL-PAYMENT-SW
          DISPLAY "*** EMPTY TERMS FILE ***"
      ELSE
          MOVE TERMS-PRINCIPAL      TO WS-BALANCE
          MOVE TERMS-INTEREST-RATE  TO WS-INTEREST-RATE
          MOVE TERMS-PAYMENT        TO WS-PAYMENT-AMOUNT
          ACCEPT WS-DATE FROM DATE
          MOVE SYSTEM-YY           TO WS-HEADING-YY
          MOVE SYSTEM-MM           TO WS-HEADING-MM
          MOVE SYSTEM-DD           TO WS-HEADING-DD
          PERFORM 600-PRODUCE-HEADINGS
      .

 200-PRODUCE-DETAIL-LINE.

      ADD 1 TO WS-PAYMENT-NUMBER
      COMPUTE WS-INTEREST-AMOUNT =
          (WS-BALANCE * WS-INTEREST-RATE) / 12
      SUBTRACT WS-INTEREST-AMOUNT FROM WS-PAYMENT-AMOUNT
          GIVING WS-PRINCIPAL-AMOUNT
      IF WS-PRINCIPAL-AMOUNT NOT LESS THAN WS-BALANCE
          MOVE "YES" TO WS-FINAL-PAYMENT-SW
      .
      IF REGULAR-PAYMENT
          MOVE WS-PAYMENT-NUMBER   TO WS-DETAIL-NUMBER
          MOVE WS-BALANCE          TO WS-DETAIL-OLD-BAL
          MOVE WS-PAYMENT-AMOUNT   TO WS-DETAIL-PAYMENT
          MOVE WS-PRINCIPAL-AMOUNT TO WS-DETAIL-PRINCIPAL
          MOVE WS-INTEREST-AMOUNT  TO WS-DETAIL-INTEREST
          SUBTRACT WS-PRINCIPAL-AMOUNT FROM WS-BALANCE
              GIVING WS-DETAIL-NEW-BAL
                     WS-BALANCE
          PERFORM 500-PRINT-LINE


 300-PRODUCE-FINAL-PAYMENT.

      MOVE WS-PAYMENT-NUMBER        TO WS-DETAIL-NUMBER
      MOVE WS-BALANCE               TO WS-DETAIL-OLD-BAL
                                       WS-DETAIL-PRINCIPAL
      ADD WS-BALANCE WS-INTEREST-AMOUNT
          GIVING WS-DETAIL-PAYMENT
      MOVE WS-INTEREST-AMOUNT       TO WS-DETAIL-INTEREST
      MOVE ZERO                     TO WS-DETAIL-NEW-BAL
      PERFORM 500-PRINT-LINE
      .
      PERFORM 100-GET-TERMS
      PERFORM 200-PRODUCE-DETAIL-LINE
          UNTIL FINAL-PAYMENT
      PERFORM 300-PRODUCE-FINAL-PAYMENT
      PERFORM 400-TERMINATE
      STOP RUN


 100-GET-TERMS.

      MOVE "NO" TO WS-TERMS-FILE-END-SW
                   WS-FINAL-PAYMENT-SW
```

```
          MOVE ZERO TO WS-PAGE-NUMBER
                       WS-PAYMENT-NUMBER
          OPEN    INPUT   TERMS-FILE
                  OUTPUT  PAYMENT-REPORT
          READ TERMS-FILE
              AT END
                  MOVE "YES" TO WS-TERMS-FILE-END-SW
          .
          IF EMPTY-TERMS-FILE
              MOVE "YES" TO WS-FINAL-PAYMENT-SW
              DISPLAY "*** EMPTY TERMS FILE ***"
          ELSE
              MOVE TERMS-PRINCIPAL      TO WS-BALANCE
              MOVE TERMS-INTEREST-RATE  TO WS-INTEREST-RATE
              MOVE TERMS-PAYMENT        TO WS-PAYMENT-AMOUNT
              ACCEPT WS-DATE FROM DATE
              MOVE SYSTEM-YY           TO WS-HEADING-YY
              MOVE SYSTEM-MM           TO WS-HEADING-MM
              MOVE SYSTEM-DD           TO WS-HEADING-DD
              PERFORM 600-PRODUCE-HEADINGS


      200-PRODUCE-DETAIL-LINE.

          ADD 1 TO WS-PAYMENT-NUMBER
          COMPUTE WS-INTEREST-AMOUNT =
              (WS-BALANCE * WS-INTEREST-RATE) / 12
          SUBTRACT WS-INTEREST-AMOUNT FROM WS-PAYMENT-AMOUNT
              GIVING WS-PRINCIPAL-AMOUNT
          IF WS-PRINCIPAL-AMOUNT NOT LESS THAN WS-BALANCE
              MOVE "YES" TO WS-FINAL-PAYMENT-SW
          .
          IF REGULAR-PAYMENT
              MOVE WS-PAYMENT-NUMBER    TO WS-DETAIL-NUMBER
              MOVE WS-BALANCE           TO WS-DETAIL-OLD-BAL
              MOVE WS-PAYMENT-AMOUNT    TO WS-DETAIL-PAYMENT
              MOVE WS-PRINCIPAL-AMOUNT  TO WS-DETAIL-PRINCIPAL
              MOVE WS-INTEREST-AMOUNT   TO WS-DETAIL-INTEREST
              SUBTRACT WS-PRINCIPAL-AMOUNT FROM WS-BALANCE
                  GIVING WS-DETAIL-NEW-BAL
                         WS-BALANCE
              PERFORM 500-PRINT-LINE
          .

      300-PRODUCE-FINAL-PAYMENT.

          MOVE WS-PAYMENT-NUMBER        TO WS-DETAIL-NUMBER
          MOVE WS-BALANCE               TO WS-DETAIL-OLD-BAL
                                           WS-DETAIL-PRINCIPAL
          ADD WS-BALANCE WS-INTEREST-AMOUNT
              GIVING WS-DETAIL-PAYMENT
          MOVE WS-INTEREST-AMOUNT       TO WS-DETAIL-INTEREST
          MOVE ZERO                     TO WS-DETAIL-NEW-BAL
          PERFORM 500-PRINT-LINE


      400-TERMINATE.

          CLOSE   TERMS-FILE
                  PAYMENT-REPORT
```

```
500-PRINT-LINE.

    WRITE PAYMENT-RECORD
        FROM WS-DETAIL-LINE
        AFTER ADVANCING 2 LINES
        AT END-OF-PAGE
            PERFORM 600-PRODUCE-HEADINGS
    .

600-PRODUCE-HEADINGS.

    ADD 1 TO WS-PAGE-NUMBER
    MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE-NO
    WRITE PAYMENT-RECORD
        FROM WS-HEADING-LINE-1
        AFTER ADVANCING PAGE
    WRITE PAYMENT-RECORD
        FROM WS-HEADING-LINE-2
        AFTER ADVANCING 2 LINES
```

شكل ( ٧ - ٢٤ )

| | MORTGAGE PAYMENTS | 5/09/83 | PAGE 1 | | |
|---|---|---|---|---|---|
| PAYMENT # | OLD BALANCE | PAYMENT | PRINCIPAL | INTEREST | NEW BALANCE |
| 1 | 6,000.00 | 1,000.00 | 950.00 | 50.00 | 5,050.00 |
| 2 | 5,050.00 | 1,000.00 | 957.92 | 42.08 | 4,092.08 |
| 3 | 4,092.08 | 1,000.00 | 965.90 | 34.10 | 3,126.18 |
| 4 | 3,126.18 | 1,000.00 | 973.95 | 26.05 | 2,152.23 |
| 5 | 2,152.23 | 1,000.00 | 982.07 | 17.93 | 1,170.16 |
| 6 | 1,170.16 | 1,000.00 | 990.25 | 9.75 | 179.91 |
| 7 | 179.91 | 181.40 | 179.91 | 1.49 | .00 |

شكل ( ٢٥ - ٧ )

## تمارين برمجة

٧ - ٩٤ بمعرفة سجلات البيع التالية الموجودة على ملف بطاقات .

| *Field* | *Data type & length* |
|---|---|
| sales ID | X(5) |
| closing date | X(6) in the form yymmdd |
| amount this year | S9(4)V99 |
| amount last year | S9(4)V99 |
| unused | 57 bytes |

اطبع تقريرا يحتوى على عناوين مناسبة ، وسطرين فقط :

(١) تعريف المبيعات ، والكمية هذا العام ، والكمية العام الماضى ، والزيادة عن العام الماضى ، كل هذا للبائع الذى لديه أقصى زيادة فى المبيعات .

(٢) نفس المعلومات لكنها للبائع الذى حقق أقل زيادة فى المبيعات .

الزيادة فى المبيعات معرفة على أنها = الكمية هذا العام ، مطروحا منها كمية العام الماضى . من الممكن فى (1) ، ومن المحتمل فى (2) أن تكون الزيادة فى المبيعات سالبة ، أى أن هناك قلة فى المبيعات .

٧ - ٩٥ عدل التمرين رقم ٩٤ بحيث يطبع البرنامج إجمالى مبيعات هذا العام ، وإجمالى مبيعات العام الماضى ، ومتوسط مبيعات هذا العام ، ومتوسط مبيعات العام الماضى . اجعل مخرجاتك واقعية وقابلة للاستخدام بقدر الإمكان .

٧ - ٩٦ عدل التمرين رقم ٩٣ بحيث يطبع البرنامج ، بعد كل الشهر الثانى عشر ، إجمالى ما دفع فى الأصل، وإجمالى ما دفع فى الفائدة أثناء فترة الاثنى عشر شهراً السابقة . كما يجب أن يطبع كذلك إجمالى الفائدة الموزعة على القرض فى نهاية التقرير.

٧ - ٩٧ اكتب برنامجا للتأكد من صحة ( التنقيح ) ، ملف مدخلات التمرين رقم ٩٤ يجب أن يطبع البرنامج تقريرا ، يسرد سجلات الملف التى تحتوى على حقول غير عددية فقط . يجب وضع خط تحت الحقول غير الصحيحة فى التقرير . تذكر إنه يمكن أن يكون هناك حقل واحد ، أو 2 أو 3 ، أو الأربعة .. كلها غير صحيح فى نفس السجل . اطبع عناوين مناسبة للتقرير ومعها تاريخ التنفيذ ، ورقم الصفحة وعناوين الأعمدة . اختبر برنامجك بدقة لكل مجموعات الخليط الممكنة لسجلات المدخلات .

# الفصل الثامن

# إعداد البرنامج :

# منهج الأجزاء ، ومن القمة إلى القاعدة

# Program Development: The Top-Down, Modular Approach

حتى الآن ترك القارئ لوسائله الخاصة بكيفية مواجهة مشكلة البرمجة . يقدم هذا الفصل عملية تطوير برامج فعالة ، سبق اختبارها ، والتى يمكن أن تفيد المبتدئين والمهنيين كذلك .

## ٨ - ١ الخطوة الأولى : تعريف المشكلة

يجب على المبرمج أن يكتسب ويتعلم كل مواد التصميم التى أنتجها محللو النظم الذين تعاملوا مع المشكلة . مثل هذه المواد يشمل خرائط تخطيطات السجلات ، وخرائط مسافات الطابع ، والسرد الوصفى بالإنجليزية للبرامج . ويجب أن يعد المبرمج ( أو المحلل ) مثل هذه المواد بنفسه .

## ٨ - ٢ الخطوة الثانية : تصميم عام للبرنامج

الخطوة التالية هيإنتاج تصميم أجزاء من أعلى لأسفل ( من القمة الى القاعدة ) للبرنامج وتوثيقه بخريطة هيكل -Struc ture Chart ( تسمى كذلك جدولاً مرئياً للمحتويات Visual table of contents (VTOC) أو خريطة هرمية (hierarchy chart) . ويسمى التصميم بأنه تصميم أجزاء modular لأنه يجزىء البرنامج إلى قطع ، والأجزاء modules ، يؤدى كل منها وظيفة معرفة تعريفا جيدا ، ويمكن كتابة كل جزء كمقطع كوبل . ويسمى التصميم من القمة إلى القاعدة top-down ؛ لأنه يركز على وظائف المستوى الاعلى أولا ( الشاملة أو المتحكمة ) التى يجب أن يؤديها البرنامج ، ويعامل وظائف المستوى الأدنى ( المحددة التفصيلية ) فيما بعد .

**مثال ٨ - ١ :**

الصورة النهائية لخريطة الهيكل لبرنامج تقرير مبيعات مبينه فى شكل ٨ - ١ . إذا كانت كتابة كل جزء (كل مستطيل ) كمقطع كوبل .. فتحدد ، الخطوط الواصلة .. أى المقاطع تنفذ أى مقاطع أخرى . لاحظ أن كل جزء يحمل اسمًا وصفيًا يعرف

وظيفة ورقم الجزء ورقم تعريف الخليط المستخدم شرطه لرقم الجزء، واسمه يجب ألا يزيد طوله عن 30 رمزا بالنسبة لاسم مقطع الكوبل . لاحظ الركن المظلل الذى يعرف جزءاً ( مقطعاً ) يسمى مُنَفَّذ بواسطة أكثر من جزء واحد من أجزاء الخريطة مثل 230 ، الذى يستدعى بواسطة كل من 100 و 220 . بالطبع يظهر PRINT - HEADINGS - 230 مرة واحدة فقط فى برنامج مصدر الكوبل ، ونستخدم أجزاء مكررة فى خريطة الهيكل لحفظ الخريطة فى شكل هرمى ( شجرى ) .

ماذا جرى فى عقل المبرمج الذى أعد شكل ٨ - ١ جزءا جزءا ، من القمة إلى القاعدة ؟

١ - المشكلة الحالية هى تقرير مبيعات طبقا للمنطقة ، وعلى هذا .. فإننى أضع جزءا فى قمة خريطة الهيكل ، وأضع له الرقم 000 ، واسميه Produce Sales by Region.

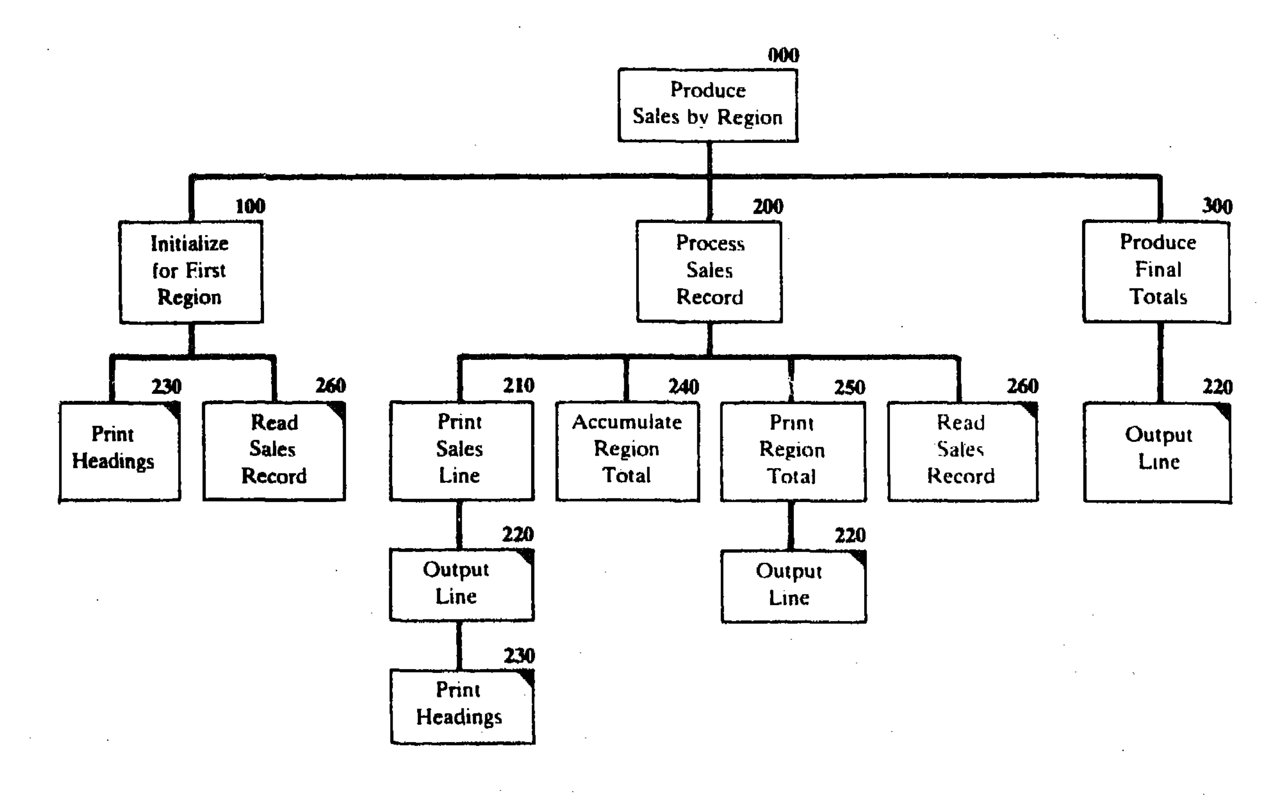
| 000 |
|---|
| Produce Sales by Region |

٢ - والآن .. ما الوظائف التى يجب أن تؤدى لكى يتم إنتاج تقرير مبيعات طبقا للمنطقة ؟ يبدو لى أن هناك ثلاث خطوات رئيسية : إعداد البرنامج لتشغيل بيانات لأول منطقة ، وتشغيل سجلات مبيعات فردية ، ومعاملة الإجماليات النهائية عند نهاية العمل . ( لا توجد طريقة صحيحة وأخرى خطأ لتجزئة المشكلة إلى أجزاء فردية ، و بالرغم من أن بعض الحلول أفضل من البعض الآخر ، فلا يوجد بصفة عامة حل واحد أفضل ) . بعد تعريف الوظائف الثلاثة الرئيسية التى تؤدى بواسطة 000 - PRODUCE - SALES - BY - REGION ، وفصل هذه الوظائف فى أجزاء خاصة بها ، تشبه خريطة الهيكل شكل ٨ - ٢ .

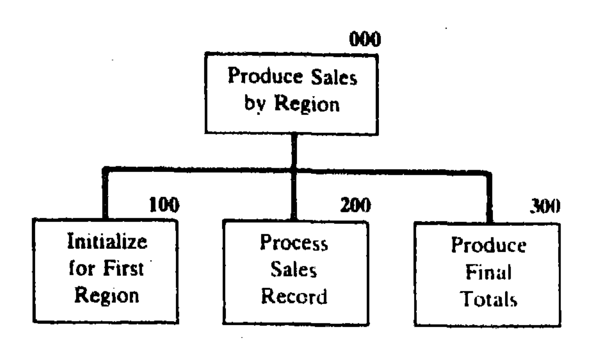
٣ - بتجزئة الجزء على المستوى 0 إلى وظائف تحكم منطقية رئيسية للبرنامج ، فإننى أكرر نفس الشىء على كل جزء على المستوى 1 . أى الوظائف المعرفة مستقلة ، والتى يجب أن تؤدى لبدء تشغيل أول منطقة ؟ حسنا .. على البرنامج أن يطبع عناوين لبدء التقرير ، ويجب كذلك أن يدخل أول سجل مبيعات ليذكر أى المناطق هى الاولى . وبالمثل .. لتشغيل سجل مبيعات ، يجب على البرنامج أن (١) يطبع سطر مبيعات فى التقرير ، ساردا محتويات سجل المبيعات (٢) هذا ويرسم إجمالى المنطقة بإضافة كمية المبيعات من سجل المبيعات هذا (٣) ويطبع ، إذا كانت هناك حاجة لذلك ، إجمالى المنطقة (إذا ما تغيرت المنطقة ) .

٤ - ويدخل سجل المبيعات التالى . وعلى هذا .. يصبح عندى أجزاء على المستوى 2 ( الأجزاء 230 و 260 و 210 و 240 و 250 و 260 و220 ) . أقوم الآن بتجزئة هذه الأجزاء على المستوى ٢ الى وظائف جزئية لإنتاج أجزاء أخرى على المستوى 30 ، وهكذا حتى لا تحتوى الأجزاء على جزئية تافهة .

تخدم خريطة الهيكل الكاملة كنموذج لتنظيم جزء الإجراءات للبرنامج . وقد أوضحت الخبرة أن محاولة توفير الوقت بترك هذه المرحلة من التصميم – أو محاولة كتابة الشفرة أولا ثم عمل التصميم ثانيا – تنتهى دائما وبكل تأكيد بوقت أطول ، وتكلفة أكثر فى برامج بها أخطاء وغير متكافئة .




شكل ( ٨ - ١ )




شكل ( ٨ - ٢ )

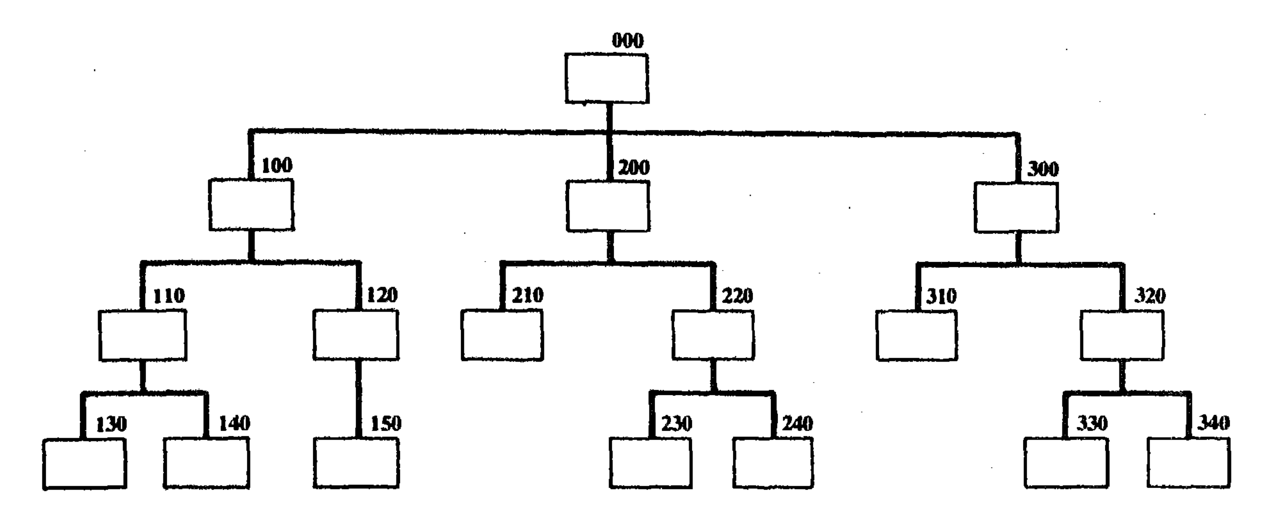
## خطوط إرشادية عند عمل خرائط الهيكل

١ - وظيفة واحدة للجزء الواحد .

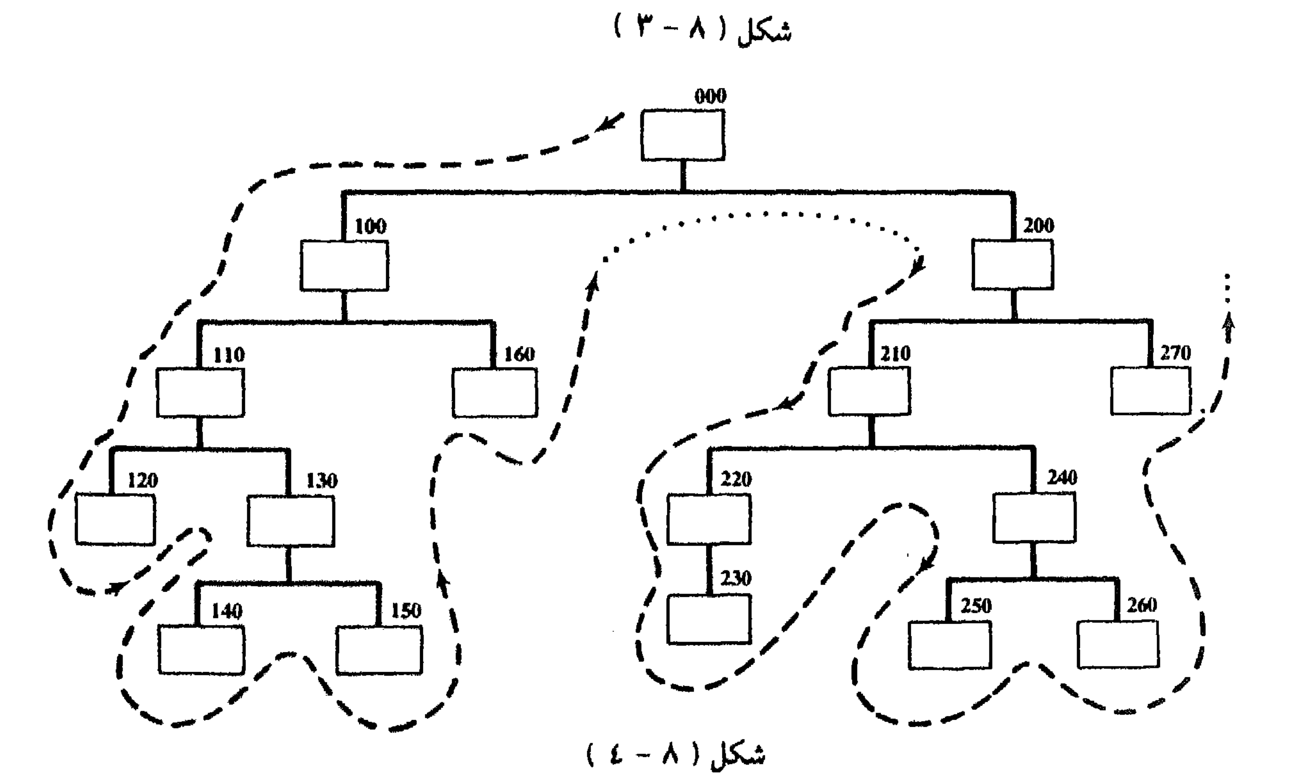
٢ - ضع رقماً لكل جزء . يقدم مثال ٨ - ٢ ومثال ٨ - ٣ طريقتين لوضع الأرقام . إذا ظهر أحد الاجزاء أكثر من مرة واحدة

فى خريطة الهيكل ( أى إنه إذا استدعى بواسطة أكثر من جزء واحد ) ، حدده فى كل مكان بالرقم الذى يصاحب حدوثه الأكثر منطقية أو الأكثر تكرارا . كذلك يجب ان تحدد الأجزاء المكررة بصفة خاصة ( كما فى شكل ٨ - ١ ) .

**مثال ٨ - ٢ :** فى الترقيم الأفقى horizontal numbering ( شكل ٨ - ٣ ) وضعت رقم 000 للمستوى 0 . ووضعت الأرقام 100, 200, 300 , ... للمستوى 1 . سلالات 100 ترقم مستوى بعد مستوى 110, 120, 130 ,... وكذلك سلالات 200 ترقم مستوى بعد مستوى 210, 220, 230 ,... الخ . لاحظ أن الترقيم يمكن تحقيقه من القمة إلى القاعدة ، مع تطوير الهيكل .




شكل ( ٨ - ٣ )




شكل ( ٨ - ٤ )

**مثال ٨ - ٣ :**

فى الترقيم الرأسى vertical numbering ( شكل ٨ - ٤ ) لا يتقدم الشخص مستوى بمستوى ، لكنه يدور لليسار ( كما يبدو فى الخريطة ) عند كل وصلة junction .

وعلى هذا .. يرقم المستوى 0 بالرقم 000 والدوران لليسار ، لمقابلة الأجزاء 100 و 110 و 120 ... وفى نهاية السطر تحدث عودة إلى آخر وصلة لأخر أول ما ترك دون أخذه . استمر فى الترقيم بزيادة 10 حتى تستهلك كل سلالة 100 . عند ذلك ابدأ الرقم 200 ( الاول من ناحية اليسار 000 لم يؤخذ حتى الآن ) ، استخدم نفس القاعدة فى ترقيم سلالات 200 - وهكذا .

والطريقة السهلة لعمل ذلك هى رسم منحنى مستمر على حافة محتويات الخريطة ( أنظر شكل ٨ - ٤) ، عند ذلك تستطيع عمل الترقيم عبر هذا المنحنى .

الترقيم الرأسى 2 الذى يميل إلى حفظ الأجزاء المنادية ، والأجزاء المنادى عليها قريبة من بعضها البعض ، يوصى باستخدامه فى نظم التخزين الافتراضى virtual storage ( افحص دليل المورد المتاح لك ) .

٣ - اعمل أجزاء متماسكة وظيفيا functionally cohesive ، والتى تعنى أن كل عبارة فى الجزء تسهم بطريقة مباشرة فى أداء وظيفة فردية للجزء .

٤ - اعمل أجزاء منخفضة فى التقارن Coupling ، وبمعنى آخر .. فى كلمات أخرى ، حاول أن تقلل عدد عناصر البيانات التى يتم الاتصال بها بمقطعين كوبل أو أكثر .

٥ - يجب ألا يتطلب الجزء أكثر من 50 سطرا من شفرة الكوبل ( أى صفحة واحدة من قائمة برنامج المصدر ) .

يجب أن يتم تقويم أى جزء أكبر من صفحة واحدة لمعرفة ما إذا كان يجب نقل بعض عملياته إلى أجزاء على مستوى أقل تستدعى بواسطة هذا الجزء أم لا.

٦ - يجب أن تستدعى الاجزاء أجزاء أقل منها فى المستوى .

٧ - جزء التنفيذ executive module الموجود على المستوى 0 يكون مسئولاً بصفة عامة عن تكرار استدعاء نموذج على المستوى 1 ) ، وهذا يعد دورة البرنامج الرئيسية . وعادة ما تكون هذه الدورة من نوع « نفذ حتى نهاية الملف » .

٨ - حاول أن تستخدم جزء READ واحداً فقط للف المدخلات ، وجزء WRITE واحداً فقط لملف المخرجات ، يمكن استدعاء جزء READ أو جزء WRITE المناسب بواسطة أى جزء ، عندما تكون هناك حاجة الى عملية مدخلات أو عملية مخرجات .

هناك استثناء يحدث فى حالة ملفات الطباعة ، والتى يمكن لاحد الاجزاء أن يكتب WRITE عناوين ، وجزء آخر يكتب سطوراً تفصيلية .

٩ - إذا ما أصبح أحد الأجزاء التى لا تتكرر تافها ( مثل وجود قلة من العبارات فيه ) اعتبر حذفه ونقل عباراته إلى جزء آخر يستدعيه .

## ٨ - ٣ الخطوة الثالثة : تصميم تفصيلى للبرنامج

بعد إنتاج نموذج عام للبرنامج فى صورة خريطة هيكل (VTOC) تصبح الخطوة التالية فى إعداد البرنامج ، هى تصميم المنطق التفصيلى لكل جزء . يمكن عمل ذلك باستخدام خرائط مسار مرتبة أو شفرة شبيهة ، كما سبق ذكره فى الفصل السابع ، إلا أنه توجد وسيلة أكثر فائدة ، وهى خريطة هرمية المدخلات والتشغيل والمخرجات hierarchical input - process - output chart ، والتى تعرف بخريطة هيبو HIPO . وتدمج صفحة فردية من خريطة هيبو ( شكل ٨ - ٥) وصف الشفرة الشبيهة للمنطق التفصيلى لأحد الأجزاء ، مع تحديد المدخلات أو عناصر بيانات مخزن العمل التى يحتاجها الجزء ، وتحديد المخرجات أو عناصر بيانات مخزن العمل التى ينتجها أو يغيرها الجزء .

يجب ملاحظة أن أحد عناصر البيانات يمكن أن يظهر فى كل من مجموعة المدخلات ومجموعة المخرجات ( اذا كان مستخرجا بواسطة الجزء كما ان الجزء ، يجرى تغييرا عليه ) . تعد صفحات هيبو المختلفة من VTOC فى صورة من القمة الى القاعدة . انظر شكل ٨ - ١١ لرؤية خريطة هيبو فعلية .

عند اتمام خريطة هيبو .. يمكن اختبار الشفرة الشبيهة الموجودة فى كل مجموعة مكتبيا لتحديد وتصحيح ما يمكن من الأخطاء قبل بدء كتابة واختبار الشفرة

## ٨ - ٤ الخطوة الرابعة : إعداد خطة لكتابة الشفرة وإجراء الاختبارات

يمكن ، ويجب ، أن يطبق منهج من القمة إلى القاعدة فى كتابة واختبار البرنامج كما حدث فى تصميمه . وهذا يعنى ما يلى:

١ - كتابة شفرة الأجزاء مرتفعة المستوى أولا ، بعد اختبار هذه الاجزاء وتصحيحها ، يمكن كتابة الاجزاء منخفضة المستوى واختبارها .

٢ - يجب ألا يكتب الجزء إلا إذا كتبت الأجزاء المحصورة بينه وبين جزء التنفيذ (المستوى 000) واختبرت كذلك.

٣ - الجزء الذى ينادى على أجزاء أخرى لم تكتب بعد يمكن اختباره باستخدام أجزاء برنامج وهمية program stubs ، تحاكى تنفيذ الجزء المنادى عليه .

عادة ما يحتوى مثل هذا الجزء الذى يحاكى جزءاً من البرنامج على اسم المقطع تليه عبارة DISPLAY فقط تطبع رسالة تبين أن المقطع قد نفذ . وفى بعض الأحيان .. تقوم مثل هذه الأجزاء بوضع حالات المفاتيح اللازمة للجزء المنادى . ( انظر شكل ٨ - ١٢ ، الأسطر من رقم ١٤٥ حتى ١٥٨ ، وشكل ٨ - ١٣ ، الأسطر من رقم ١٩٣ وحتى ٢١٥ ) .

يفضل بعض المبرمجين كتابة شفرة الأجزاء واختبارها مستوى بمستوى ( أى فى تسلسل الترقيم الأفقى ، انظر مثال ٨-٢) . يستخدم البعض الآخر ترتيب الترقيم الرأسى ( انظر مثال ٨ - ٣) . وطالما أن القاعدة رقم 2 سالفة الذكر متبعة .. فأى خليط من هذين المنهجين يكون جيدا .

شكل ( ٨ - ٥ )

| | | |
|---|---|---|
| Designer ______ | System/Program Name ______ | Date ______ |
| Module No. ______ | Module Name ______ | |

| Input | Process | Output |
|---|---|---|
| | | |

**مثال ٨ – ٤ :**

فيما يلى خطة كتابة شفرة برنامج شكل ٨ – ١ واختباره .

| رقم التنفيذ | الأجزاء | الاختبار ب ... |
|---|---|---|
| 1 | 000 | كل الملفات فارغة مع استخدام أجزاء تحاكى 100 و 200 و 300. |
| 2 | 100, 200, 300 | بيانات منطقة مبيعات واحدة ، واستخدام أجزاء تحاكى الأجزاء ,220, 250 240, 210, 260, 230 |
| 3 | 210, 220, 230, 260, 240, 250 | بيانات لمنطقتين . |
| 4 | | ( أ ) كل الملفات فارغة . |
| | | ( ب ) بيانات كل المناطق – بما فى ذلك منطقة واحدة بها سجل واحد فقط . |
| | | ( جـ ) بيانات كافية تتسبب فى عمل سريان زائد للصفحة |

## ٨ – ٥ الخطوة الخامسة : تجميع بيانات الاختبارات

فى بعض الأحيان .. يعد المبرمج بيانات الاختبارات ، وفى بعض الأحيان الأخرى .. يمكن إنتاج هذه البيانات بواسطة برنامج خاص ، ويمكن فى بعض الأحيان الحصول عليها بنسخها من ملفات حقيقة تستخدمها برامج موجودة فعلا . عندما تجمع بيانات الاختبارات .. يجب ان يحدد المبرمج المخرجات الصحيحة المناظرة لهذه البيانات ، بحيث يمكن تعريف أى تعارضات بينها وبين المخرجات الفعلية .. ليتم تصحيح البرنامج .

وفى الحالة المثالية ، عند نهاية عملية الاختبارات .. يجب أن يتم اختبار كل مسار منطقى خلال كل جزء بصفة خاصة ، ويجب أن تقدم بيانات الأختبارات التى تأخذ كل IF خلال جزء true، بينما تستدعى بيانات اخرى جزء false . يختبر كل تكرار PERFORM ... UNTIL لعدد مرات التكرار صفر ، 1 ، 2 ... إلخ ( بإجمالى عدد مرات تكراره ) . قد يكون من المستحيل للبرامج الكبيرة تقديم بيانات كافية لاختبار كل المسارات المنطقية الممكنة فى وقت مناسب . فى هذه الحالة ، يجب اختبار المسارات الأكثر أهمية ، والاكثر استخداما على الأقل بدقة .

## ٨ – ٦ الخطوة السادسة : كتابة الشفرة وعمل الاختبارات من القمة إلى القاعدة

يجب أن يكون البرنامج مكتوبا الآن ومختبرا ، جزءا بجزء ، طبقا للتسلسل الذى أعد فى خطة كتابة الشفرة وإجراء الاختبارات .. وباستخدام بيانات الاختبارات التى أجرى عليها تحليل . انظر المسألة رقم 30 من هذا الفصل لرؤية تنفيذ هذه الخطوة .

## ٨ – ٧ الخطوة السابعة : إكمال توثيق البرنامج

بعد اختبار البرنامج وتصحيحه تماما ، يجب أن تجمع خريطة الهيكل وخريطة هيبو ، وخطة كتابة الشفرة وإجراء الاختبارات والبيانات الاختبارية وتنفيذات الاختبارات مع المخرجات وقوائم برنامج المصدر . ويجب إدخال أى تغيير داخل على البرنامج أثناء عملية الاختبار والتصحيح فى VTOC و HIPO .

يجب حفظ كل الوثائق ليستخدمها مبرمجو الصيانة المحددين لإجراء تعديلات على البرنامج فيما بعد .

## اسئلة مراجعة

٨ – ١ وضح أول خطوة فى إعداد برنامج فردى .

٨ – ٢ ما خريطة الهيكل (VTOC) ؟ كيف تستخدم فى التصميم العام للبرنامج ؟

٨ – ٣ ما معنى التصميم من أعلى لأسفل ، أو من القمة إلى القاعدة ؟

٨ – ٤ ما هو الجزء module ؟

٨ – ٥ عرف مفهوم المستوى فى خريطة الهيكل .

٨ – ٦ قارن مع المضاهاة بين طريقة الترقيم الأفقى ، وطريقة الترقيم الرأسى .

٨ – ٧ وضح أهمية القاعدة « جزء واحد لوظيفة واحدة »

٨ – ٨ لماذا يجب أن تعرض الأجزاء تماسكا مرتفعا ؟

٨ – ٩ ماهو جزء التنفيذ ؟

٨ – ١٠ ماذا يوصى بوجود عبارة READ واحدة، أو عبارة WRITE واحدة للملف المستخدم فى البرنامج ؟ كيف يمكن تحقيق ذلك ؟

٨ – ١١ وضح استخدام خريطة HIPO فى التصميم التفصيلى للبرنامج .

٨ – ١٢ ناقش ما يحدث فى مجموعات المدخلات والعمليات والمخرجات من خريطة هيبو .

٨ – ١٣ ماذا تعنى كتابة الشفرة وعمل الاختبارات من القمة الى القاعدة ؟

٨ – ١٤ ما جزء البرنامج الذى يحاكى جزءا لم تكتب شفرته ؟

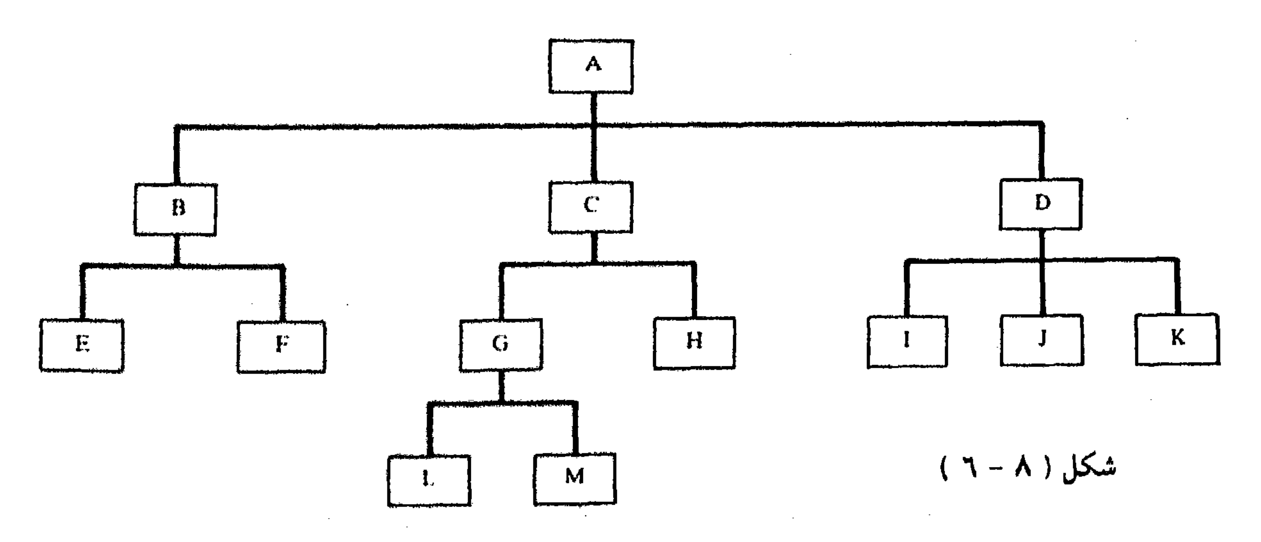
٨ – ١٥ قارن المنهج الأفقى لكتابة الشفرة وعمل الاختبارات من القمة إلى القاعدة مع المنهج الرأسى .

٨ – ١٦ ما هدف اختبار البرنامج ؟ وضح كيف يجب أن تتغير بيانات الاختبارات أثناء عملية الاختبار من القمة إلى القاعدة .

٨ – ١٧ ما أهمية توثيق البرنامج ؟ وضح كيف يمكن لوسائل تصميم البرنامج أن تعمل فيما بعد كتوثيق للبرنامج .

## مسائل محلولة

٨ – ١٨ ضع أرقاماً لأجزاء شكل ٨ – ٦ ( أ ) أفقيا (ب) ورأسيا .




شكل ( ٨ – ٦ )

٨ – ١٩ هل الجزء ( مقطع الكوبل ) التالى قوى التماسك ؟

```
SAMPLE-PARA.
    MOVE SPACES TO SALARIED-MESSAGE-AREA
    MOVE ZEROS  TO TOTAL-SALARY-AMOUNT
    PERFORM READ-TIME-CARD
    ADD TIME-CARD-HOURS TO TOTAL-HOURS
    IF TIME-CARD-ID NOT NUMERIC
        MOVE "YES" TO INVALID-SWITCH

    PERFORM COMPUTE-TAX
    IF LINE-COUNT-ON-REPORT GREATER THAN 30
        PERFORM PRINT-HEADINGS
```

الجزء المتماسك هو الجزء الذى تسهم كل عبارة فيه فى أداء وظيفة معرفة جيدا . الجزء الذى (١) يضع قيماً ابتدائية لبعض عناصر البيانات (٢) ويدخل سجلاً (٣) ويرسم إجماليات (٤) ويتحقق من صحة حقل (٥) ويحسب ضريبة الدخل (٦) ويعامل عناوين الطباعة عند الوصول إلى نهاية الصفحة ، يكون بالتأكيد ذا تماسك منخفض للغاية .

٨ – ٢٠ ماهى الوظائف التى يؤديها جزء التنفيذ فى أحد تطبيقات الاعمال التقليدية ؟

عادة ما يؤدى جزء التنفيذ أربع وظائف .

```
SAMPLE-EXECUTIVE-MODULE.
    PERFORM INITIALIZATION-MODULE
    PERFORM MAIN-PROCESSING-MODULE
        UNTIL END-OF-FILE
    PERFORM TERMINATION-MODULE
    STOP RUN
```

يجعل الجزء INITIALIZATION - MODULE البرنامج يبدأ ( تفتح الملفات ، توضع أصفار فى العدادات ، وفى مناطق الإجماليات ، يقرأ أول سجل منطقى .. إلخ ) . الجزء TERMINATION - MODULE يحضر البرنامج إلى تعليق مرتب ( طباعه إجماليات نهائية ونهايات التقرير ، وإغلاق الملفات ... إلخ ) . الجزء MAIN - PROCESSING - MODULE يؤدى الغرض الرئيسى من البرنامج ( مثل طباعة شيكات ، والتأكد من صحة ملف المدخلات ، وتجديد الملف الرئيسى وطباعة التقرير ) . إذا شملت عملية الإعداد initializatian والإنهاء termination عملا بسيطا نسبيا ، فلا تحتاج أن تنفذ كأجزاء مستقلة ويمكن كتابتها داخل جزء التنفيذ نفسه .

٨ - ٢١ انقذ معالجة READ فيما يلي :

```
100-INITIALIZE-PROGRAM
    MOVE ZERO TO...
    MOVE SPACES TO...
    OPEN...
    READ INPUT-FILE
        AT END  MOVE "YES" TO END-FILE-SW

    IF END-FILE-SW NOT EQUAL "YES"
        MOVE A TO B

......................
210-PRODUCE-REPORT-LINE.
    MOVE...
    ADD...
    READ INPUT-FILE
        AT END  MOVE "YES" TO END-FILE-SW
```

هنا استخدمت عبارة READ منفصلة كلما كان مطلوبا إدخال سجل منطقى . و يمكن أن يكون هذا غير مقنع إذا تغيرت مواصفات المدخلات والمخرجات للبرنامج ؛ حيث يجب على مبرمج الصيانه أن يتصيد - خلال محتويات البرنامج - كل عبارة READ . هناك طريقة أفضل ، وهى وضع عبارة READ للملف فى مقطع خاص بها :

```
100-INITIALIZE-PROGRAM.
    MOVE ZERO TO...
    MOVE SPACES TO...
    OPEN...
    PERFORM GET-INPUT-RECORD
    IF END-FILE-SW NOT EQUAL "YES"
        MOVE A TO B

................................
210-PRODUCE-REPORT-LINE.
    MOVE...
    ADD...
    PERFORM GET-INPUT-RECORD
......................
GET-INPUT-RECORD.
    READ INPUT-FILE
        AT END
            MOVE "YES" TO END-FILE-SW
```

٨ – ٢٢ أحد أجزاء شكل ٨ – ١ تافها بالتأكيد ، عرفه .

يحتوى TOTAL - REGION - ACCUMULATE - 240 على عبارة ADD واحدة ، يمكن إدخالها ببساطة فى الجزء المنادى ، 200 - PROCESS - SALES - RECORD

٨ – ٢٣ لماذا يجب أن تبدأ بيانات الاختبارات بسيطة ؟

البرنامج الجديد به عديد من الأخطاء بصفة عامة . وفى البداية تتسبب بيانات الاختبارات المعقدة فى تداخل الأخطاء مع بعضها البعض بطرق ، يمكن أن تكون مضلله وتجعل الأخطاء صعبة التعريف . أكثر من ذلك فى عملية الاختبار ، عندما تصحح معظم الاخطاء .. فإنه يمكن استخدام بيانات اختبارات أكثر تعقيدا .

٨ – ٢٤ افرض أن C, B, A هى حقول فى سجل مدخلات .

حدد مجموعة من قيم اختبارية لكل من C, B, A تختبر كل المسارات المنطقية فى الشفرة التالية :

```
IF A LESS THAN B
    MOVE...
    ADD...
    IF A LESS THAN C
        PERFORM...
    ELSE
        PERFORM...
ELSE
    SUBTRACT...
    IF B EQUAL C
        MOVE...
    ELSE
        PERFORM...
        IF A EQUAL C
            MOVE...
```

ارسم شجرة قرارات decision tree لعبارات IF المتداخلة ، شكل ٨ – ٧ ، المسارات المنطقية معطاة بالمسارات من الجذر A < B إلى الأوراق أو نقاط النهاية للشجرة . وتبدو هناك 5 مسارات ، موضح بها بيانات اختبارية فى جدول ٨ – ١

جدول ( ٨ – ١ )

| Path | A | B | C |
|---|---|---|---|
| (1) | 1 | 2 | 3 |
| (2) | 1 | 2 | 0 |
| (3) | 2 | 1 | 1 |
| (4) | 2 | 1 | 2 |
| (5) | 2 | 1 | 3 |



شكل ( ٨ – ٧ )

٨ – ٢٥ تصل عديد من برامج إعداد تقارير الأعمال إلى معلوماتها عن طريق تشغيل ملفات على شرائط أو أقراص ، تبنى هذه البرامج حول دورة رئيسية ، يتم إدخال سجل منطقى فيها وتشغيله ، وإذا كان مناسبا .. يتم إخراج النتائج . هناك منهجان أساسيان لتصميم المنطق لمثل هذا البرنامج . فى المنهج الاول .. يدخل جسم الدورة الرئيسية (١) سجلا منطقيا (٢) يختبر نهاية الملف (٣) إذا لم تحدث نهاية الملف .. يقوم بتشغيل السجل المنطقى الذى تم إدخاله على التو .

أكتب شفرة شبيهة لتوضيح هذا المنطق للبرنامج الذى يسرد الرواتب الشهرية للعاملين بعد قراءته الملف الرئيسى للعاملين .

```
set end-of-file-switch "off"
open employee-master, report-file
perform until end-of-file-switch is "on" (main loop):
    read employee-master-record
        at end:   set end-of-file-switch "on"
    if end-of-file-switch is "off"
        perform process-employee-record
        perform output-employee-results
    endif
endperform
close employee-master, report-file
stop
```

اختبار نهاية الملف حرجة فى هذا المنهج حيث إنها تتم قبل تشغيل السجل المنطقى ( انظر مثال ٦ – ١٠ ) . فى بعض النظم .. لاتزال منطقة السجل المنطقى (FD) تحتوى آخر سجل تمت قراءته عندما حدثت نهاية الملف . وفى هذه الحالة يمكن أن يؤدى الفشل فى اختبار نهاية الملف قبل تشغيل سجل إلى تشغيل آخر سجل مرتين.

٨ – ٢٦ المنهج الأساسى الثانى لنفس نوع البرامج التى سبق مناقشتها فى المسألة السابقة يستخدم قراءة أولية priming read، وهى قراءة أولية للملف المنفذ قبل بدء دورة البرنامج الرئيسية . وحيث إنه تم إدخال سجل منطقى فعلا عندما تنفذ الدوره الرئيسية .. فإن أول شىء تؤديه الدورة الرئيسية هو (١) تشغيل سجل منطقى واخراج النتائج ، عند ذلك (٢) تقرأ السجل المنطقى التالى ( الذى يجرى عليه تشغيل عندما تتكرر الدورة عند الخطوة الأولى ) . لمنهج القراءة الأولية ميزتان :

(١) عدم وجود الحاجة إلى اختبار نهاية الملف داخل الدورة .

(٢) حيث إن أول سجل تم إدخاله منفصلا .. فمن الممكن أن يعالج معالجة خاصة ( انظر المسألة التالية ) . أعد كتابة المسألة السابقة مع استخدام القرائة الاولية .

```
set end-of-file-switch "off"
open employee-master, report-file
read employee-master record (priming read)
    at end:   set end-of-file-switch "on"
... (any special processing required for the first
    logical record could be done here)
perform until end-of-file-switch is "on" (main loop):
    perform process-employee-record
    perform output-employee-results
    read employee-master record
        at end:   set end-of-file-switch "on"
endperform
close employee-master, report-file
stop
```

٨ – ٢٧ بين كيف يمكن استخدام منهج المسألة رقم ٢٥ من هذا الفصل ( الذى لا يستخدم قراءة أولية ) إذا كان أول سجل منطقى فى الملف يتطلب تشغيلا خاصا .

```
set end-of-file-switch to "off"
set first-time-switch to "on"
open employee-master, report-file
perform until end-of-file-switch is "on" (main loop):
    read employee-master record
        at end:  set end-of-file-switch "on"
    if end-of-file-switch is "off"
        if first-time-switch is "on"
            perform first-record-only-processing
            perform first-record-only-output
            set first-time-switch "off"
        else
            perform process-employee-record
            perform output-employee-results
        endif
    endif
endperform
close employee-master, report-file
stop
```

٨ – ٢٨ أسلوب القراءة الاولية مفيد بصفة خاصة فى مشاكل التحكم المتقطع control break problems ، حيث تخزن السجلات المنطقية للملف فى حقل تحكم control field واحد أو أكثر من حقل تحكم واحد . عندما تختلف القيمة الموجودة فى حقل التحكم للسجل المنطقى الحالى عن القيمة الموجودة فى حقل التحكم المتقطع للسجل المنطقى السابق ، يقال إنه حدث تحكم متقطع control break . عند هذه النقطة .. ويمكن استدعاء تشغيل خاص . مثال ذلك ، افرض أن ملف المسألة ٢٥ من هذا الفصل يشمل رقم القسم لكل عامل ، وأن الملف مرتب ترتيبا تصاعديا طبقا لرقم القسم . اطبع تقريرا للملف لايبين الرواتب الفردية فقط ، بل يبين كذلك إجمالى الراتب لكل قسم ، وإجمالى نهائى للأقسام كلها .

```
set eof "off" (end of file switch)
open files
zero:  department-total, grand-total
read employee-master record
    at end:  set eof "on"
if eof is "off":
    move current-department-# to previous-department-#
endif

perform until eof is "on" (main loop):
    if current-department-# not equal previous-department-#
    (this is a control break)
        perform print-department-total
        add department-total to grand-total
        move zero to department-total
```

```
            move current-department-# to previous-department-#
      endif
      perform print-current-employee-salary
      add current-employee-salary to department-total
      read employee-master record
            at end:   set eof "on"
endperform
```

*(end of file should be treated as a control break to print last department total)*

```
perform print-department-total
add department-total to grand-total
perform print-grand-total
close files
stop
```

فيما يلى بعض تعليقات على مسألة التحكم المتقطع فردية المستوى Single - level ( حقل تحكم واحد ) :

(١) لقد اختبرنا نهاية الملف فى أول سجل ( القراءة الأولية ) . هذا غير مرغوب فى حدوثه ، إلا أنه ممكن ( فمثلا يمكن حدوث أخطاء فى لغة تحكم العمل ، أو خطأ من مشغل الكمبيوتر عندما ينفذ البرنامج ... إلخ ) . إن وضع رقم القسم السابق فى رقم القسم لأول سجل يقرأ يؤكد أن التحكم المتقطع لن ينفذ بطريق خطأ على أول سجل عند الدخول فى الدورة الرئيسية .

(٢) يجب إعادة إعداد إجمالى القسم بأنه صفر فى كل مرة يتغير رقم القسم (وإلا فإنه يحدث تراكم فى إجمالى القسم لإجمالى جار لكل الرواتب فى كل الأقسام ) .

(٣) بمجرد تمييز تغيير فى حقل التحكم .. يحدث التغيير التالى فى القيمة بقيمة مختلفة عن القيمة التى تسببت فى التغيير الحالى . وعلى هذا .. يجب أن يتغير رقم القسم السابق إلى رقم القسم الحالى عند كل تحكم متقطع .

(٤) يمكن إيجاد الإجمالى الشامل بكفاءة أعلى عن طريق جمع إجماليات الأقسام الفردية . يجب أن يحدث هذا عند كل تحكم متقطع .

(٥) عندما تحدث نهاية الملف .. يجب أن تعامل مثل التحكم المتقطع ؛ لانه يمكن ألا يكون هناك تغيير فى حقل التحكم لآخر رقم قسم فى الملف . وهذا يعنى أن إجمالى آخر قسم يجب أن يطبع فى نهاية الملف ، كما يجب كذلك جمع آخر إجمالى قسم على الإجمالى الشامل إذا أردنا للإجمالى الشامل أن يكون صحيحا .

٨ – ٢٩ افرض فى المسألة السابقة أن كل سجل رئيسى لعامل يحتوى على رقم وحدة division number ، وأن الملف مرتب طبقا لرقم الوحدة أولا ( حقل تحكم رئيسى ) ، ثم طبقا لرقم القسم داخل الوحدة ( حقل تحكم صغير ) . أعد تصميم منطق المسألة رقم ٢٨ لطباعة إجمالى القسم عندما تتغير أرقام الأقسام ( تحكم متقطع صغير ) ، وطباعة إجمالى الوحدة عندما يتغير رقم الوحدة ( تحكم متغير رئيسى ) . يجب أن يأخذ التقرير فى الشكل الموجود فى شكل ٨ – ٨ .

```
employee salary (division 1000, department 1)
employee salary (division 1000, department 1)
department 1 total
employee salary (division 1000, department 2)
employee salary (division 1000, department 2)
department 2 total
division 1000 total
employee salary (division 2000, department 1)
department 1 total
employee salary (division 2000, department 2)
employee salary (division 2000, department 2)
department 2 total
division 2000 total
grand total
```

شكل ( ٨ - ٨ )

```
set eof "off"
open files
zero:  department-total, division-total, grand-total
read employee-master record
     at end:  set eof "on"
if eof is "off":
     move current-department-# to previous-department-#
     move current-division-# to previous-division-#
endif

perform until eof is "on" (main loop):
     if current-division-# not equal previous-division-#
     (this is a major control break)

          perform handle-department-break
          perform print-division-total
          add division-total to grand-total
          move zero to division-total
          move current-division-# to previous-division-#
    else if current-department-# not equal previous-department-#
    (this is a minor control break)
          perform handle-department-break
    endif
    perform print-current-employee-salary
    add current-employee-salary to department-total
    read employee-master record
         at end:  set eof "on"
endperform

(end of file should be treated as a major control break)

perform handle-department-break
perform print-division-total
add division-total to grand-total
perform print-grand-total
close files
stop
```

هنا يعرف handle - department - break كما يلى:

```
handle-department-break:
    perform print-department-total
    add department-total to division-total
    move zero to department-total
    move current-department-# to previous-department-#
```

تعليقات على خوارزمى تحكم متقطع مزدوج المستوى :

١ – يجب وضع قيم ابتدائية لحقل التحكم الصغير ، وحقل التحكم الرئيسى مساوية لقيم أول سجل منطقى . يعد تحقيق هذا بسيطاً اذا استخدمت قراءة أولية .

٢ – يبدأ منطق دورة البرنامج الرئيسية باختبار التحكم المتقطع الرئيسى . لاحظ أنه بسبب وضع قيمة ابتدائية لرقم الوحدة السابقة بأنها أول رقم وحدة حالية ، فلا يتحدد تحكم متقطع لأول سجل فى الملف .

٣ – يبدأ تشغيل التحكم المتقطع الرئيسى بعمل تحكم متقطع صغير . ومن الاخطاء الشائعة الحدوث للمبتدئين اهمالهم حقيقة أنه عند تغيير حقل تحكم رئيسى ( رقم الوحدة ) يجب أن تطبع إجماليات حقل التحكم الصغير ( رقم القسم ) الأخير ، ويعاد إعداد إجمالى حقل التحكم الصغير . افحص شكل ٨ – ٨ للتاكد من فهمك سبب ذلك .

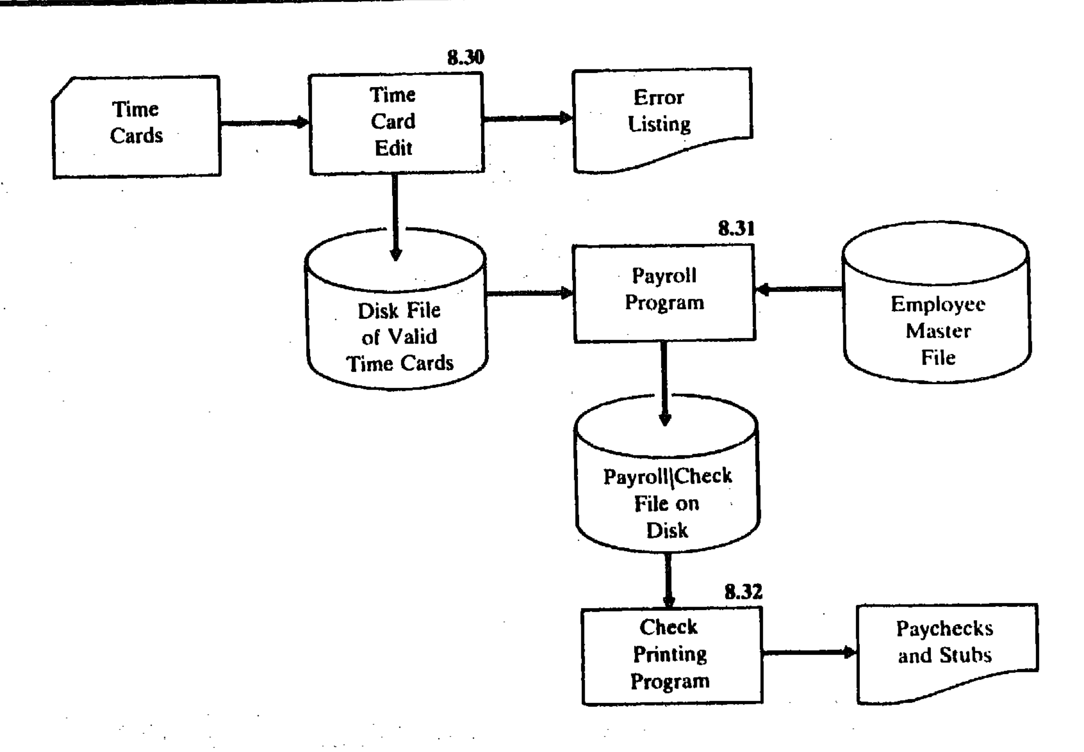
٤ – بعد معالجة تشغيل التحكم المتقطع الرئيسى للتحكم المتقطع الصغير ، فيؤدى جزء التقطع الرئيسى كل ما هو مناسب للتغيير فى حقل التحكم الرئيسى . لاحظ بصفة خاصة أن رقم الوحدة السابق أعيد إعداده بوضع رقم الوحدة الحالى فيه أثناء تشغيل التحكم المتقطع الرئيسى ، وأعيدت إجماليات حقل التحكم الرئيسى لتصبح صفرا .

٥ – بعد أن تختبر الدورة الرئيسية للتحكم المتقطع الرئيسى .. فيجب أن تختبر للتحكم المتقطع الصغير . لاحظ أن ذلك يمكن أن يحدث بهيكل IF خطى . ينفذ جزء تقطع صغير مثل مقطع PERFORM ، حيث إنه يجب استدعاؤه من جزء التقطع الرئيسى ، وجزء التقطع الصغير .

٦ – منطق كل من جزئى التحكم المتقطع هو كما يلى : (١) استدع جزء التحكم المتقطع المنخفض المستوى التالى ( اذا كان هناك مثل هذا التحكم المتقطع ) ، (٢) اطبع إجماليات تحكم لهذا التقطع ، (٣) ضف إجمالى التحكم هذا إلى إجمالى المستوى الأعلى التالى (٤) ضع صفراً فى هذا الإجمالى (٥) ضع حقل التحكم الحالى فى حقل التحكم السابق .

٧ – مرة أخرى .. يجب أن تعامل نهاية الملف كتحكم متقطع رئيسى (أولا فإن إجمالى أخر قسم واجمالى أخر وحدة لن يطبعا ولن يضافا إلى الإجمالى الشامل ) .

٨ – ٣٠ شكل ٨ – ٩ هو خريطة مسار النظام system flowchart – والتى ترسم العلاقات بين البرامج والملفات – لنظام رواتب (مبسط جدا ) . ( رموز ملف البطاقات وملف القرص واضحة ذاتيا ، وملف الطباعة يبدو كورقة مقطوعة ) . صمم واكتب شفرة برنامج تنقيح واختبره لنظام الرواتب هذا .




شكل ( ٨ - ٩ )

كما نعرف ( مثال ٦ - ٣٥ ) الغرض من برنامج التنقيح هو التأكيد من صحة المعلومات التى يقوم بإدخالها الآدميون .

يجب أن يعطى برنامج التنقيح بصفة خاصة أجوبة على الأسئلة التالية :

١ - هل الحقول العددية عددية فعلا ؟ وهل الحقول الحرفية حرفية .. فعلا ؟ ... إلخ .

٢ - هل تقع القيم داخل مدى مناسب ( مثل اليوم من الشهر يجب أن يقع بين 1 و 31 ) ؟

٣ - هل القيم الموجودة فى السجل متوافقة مع بعضها البعض ( مثل أن تاريخ الاستلام لا يمكن أن يسبق تاريخ الشحن ) ؟

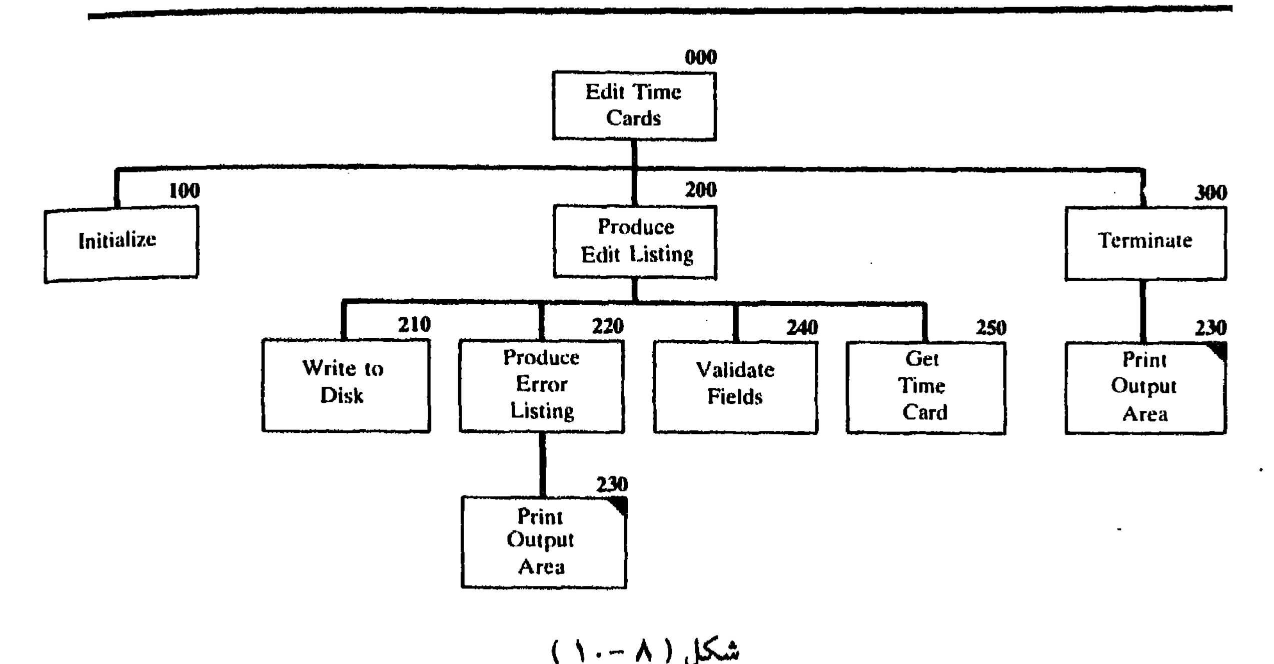
٤ - هل تحتوى الحقول المستخدم فيها رموزاً على الرموز المسموح بها فقط ؟

٥ - هل السجلات المنطقية التى يفترض أنها مخزنة ( مرتبة على التتابع ) فى حقل خاص مرتبة ترتيبا صحيحا ؟( هذا النوع من التأكد من الصحة عادة ما يسمى بالتاكد من التتابع ) .

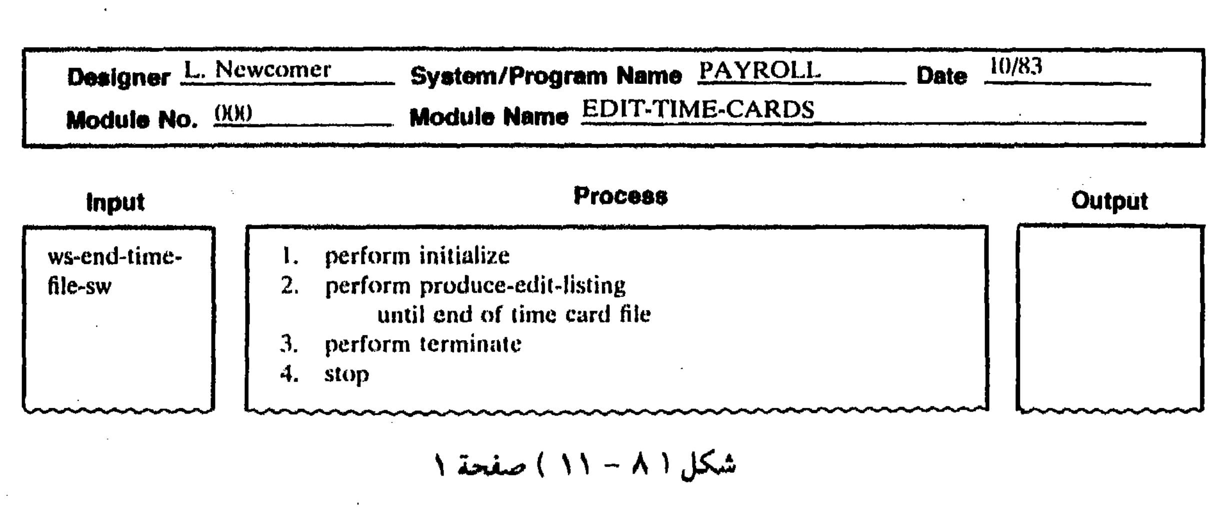
٦ - هل إجماليات تحكم الدفعة دقيقة ؟ ( حيث إن دفعات من السجلات يقوم بإدخالها مشغل الكمبيوتر ، فعادة ما تحسب إجماليات تحكم يدويا ، بجمع كل محتويات حقول محددة . عندما يقوم برنامج كمبيوتر بتشغيل الملف الناتج ، فيمكن أن يجمع إجماليًا لنفس الحقل ( الحقول ) كذلك . إذا اتفق إجمالى الكمبيوتر مع الإجمالى اليدوى ، فيفترض أن كل البيانات أجرى عليها تشغيل ) .

نرسم الآن خريطة هيكل ( شكل ٨ - ١٠ ) ، وخريطة هيبو ( شكل ٨ - ١١ ) ، وتنفيذ أول وثانى اختبار من القمة إلى القاعدة ( شكل ٨ - ١٢ وشكل ٨ - ١٣ ) ، وأخر تنفيذ ( شكل ٨ - ١٤ ) ، لبرنامج تنقيح الرواتب .

لاحظ بصفة خاصة معاملة عمل الصفحات بدون جزء LINAGE ( انظر الأجزاء , 230 - PRINT - OUTPUT - AREA ( 100 - INITIALIZES




شكل ( ٨ - ١٠ )



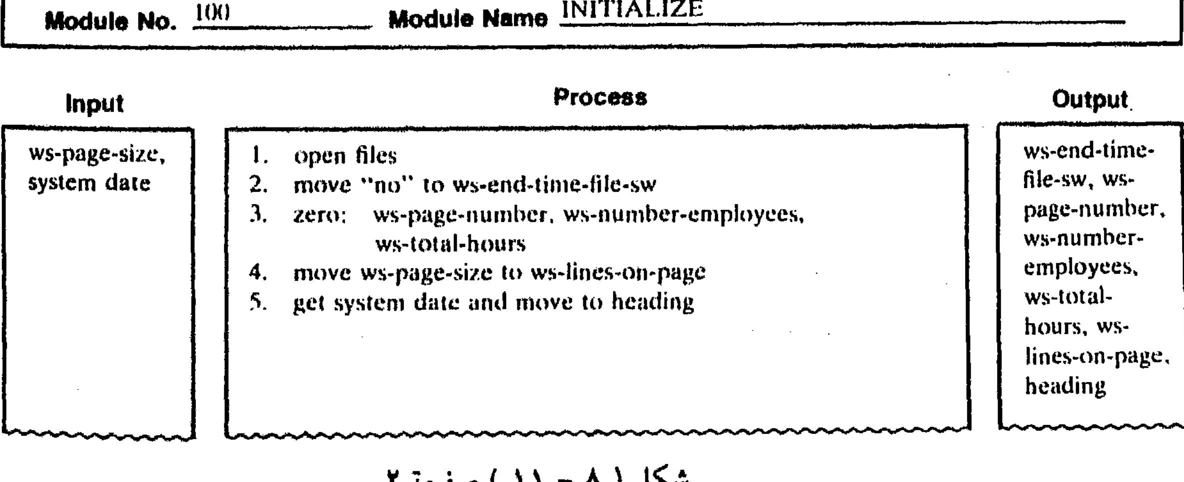
**Designer** L. Newcomer **System/Program Name** PAYROLL **Date** 10/83
**Module No.** 000 **Module Name** EDIT-TIME-CARDS

| Input | Process | Output |
|---|---|---|
| ws-end-time-file-sw | 1. perform initialize<br>2. perform produce-edit-listing<br>until end of time card file<br>3. perform terminate<br>4. stop | |

شكل ( ٨ - ١١ ) صفحة ١



**Designer** L. Newcomer **System/Program Name** PAYROLL **Date** 10/83
**Module No.** 100 **Module Name** INITIALIZE



| Input | Process | Output |
|---|---|---|
| ws-page-size, system date | 1. open files<br>2. move "no" to ws-end-time-file-sw<br>3. zero: ws-page-number, ws-number-employees,<br>ws-total-hours<br>4. move ws-page-size to ws-lines-on-page<br>5. get system date and move to heading | ws-end-time-file-sw, ws-page-number, ws-number-employees, ws-total-hours, ws-lines-on-page, heading |

شكل ( ٨ - ١١ ) صفحة ٢

**Designer** L. Newcomer **System/Program Name** PAYROLL **Date** 10/83
**Module No.** 200 **Module Name** PRODUCE-EDIT-LISTING

| Input | Process | Output |
| --- | --- | --- |
| ws-end-time-file-sw, ws-error-detected-sw, time-card-hours | 1. perform get-time-card<br>2. if a card was read<br>perform validate-fields<br>if record was valid<br>add 1 to ws-number-employees<br>add time-card-hours to ws-total-hours<br>perform write-to-disk<br>else<br>perform produce-error-listing | ws-number-employees, ws-total-hours |

شكل ( ٨ - ١١ ) صفحة ٣

**Designer** L. Newcomer **System/Program Name** PAYROLL **Date** 10/83
**Module No.** 300 **Module Name** TERMINATE

| Input | Process | Output |
| --- | --- | --- |
| ws-number-employees, ws-total hours | 1. move ws-number-employees and ws-total-hours to footing-line<br>2. perform print-output-area<br>3. close files | footing-line |

شكل ( ٨ - ١١ ) صفحة ٤

**Designer** L. Newcomer **System/Program Name** PAYROLL **Date** 10/83
**Module No.** 220 **Module Name** PRODUCE ERROR LISTING

| Input | Process | Output |
|---|---|---|
| time-card,<br>underline | 1. move all time card fields to detail-line<br>2. move detail-line to output-area<br>3. perform print-output-area<br>4. move underline to output-area<br>5. perform print-output-area | detail-line<br>output-area |

شكل ( ٨ - ١١ ) صفحة ٥

**Designer** L. Newcomer **System/Program Name** PAYROLL **Date** 10/83
**Module No.** 240 **Module Name** VALIDATE-FIELDS

| Input | Process | Output |
|---|---|---|
| time-card-id,<br>time-card-hours,<br>time-card-date,<br>time-card-department | 1. move "no" to error-detected-sw<br>2. move spaces to under-line<br>3. if time-card-id not numeric<br>set up underline for id<br>set error-detected-sw to "yes"<br>4. if time-card-hours not numeric<br>set up underline for hours<br>set error-detected-sw to "yes"<br>5. if time-card-date not numeric<br>set up underline for date<br>set error-detected-sw to "yes"<br>6. if time-card-department not numeric<br>set up underline for department<br>set error-detected-sw to "yes" | error-detected-sw,<br>underline |

شكل ( ٨ - ١١ ) صفحة ٦

| | | | |
|---|---|---|---|
| Designer L. Newcomer | System/Program Name PAYROLL | | Date 10/83 |
| Module No. 230 | Module Name PRINT-OUTPUT-AREA | | |

| Input | Process | Output |
|---|---|---|
| ws-lines-on-page, ws-number-to-skip, ws-page-size, heading-1, heading-2, output-area | 1. if ws-lines-on-page + ws-number-to-skip<br>greater than ws-page-size<br>add 1 to ws-page-number<br>move ws-page-number to ws-heading-page<br>write error-line from heading-1<br>after page<br>write error-line from heading-2<br>after 2 lines<br>move 2 to ws-lines-on-page<br>2. Write error-line<br>from output-area<br>after 2 lines<br>3. add number of lines skipped to ws-lines-on-page | ws-page-number, ws-heading-page, error-line, heading-1, heading-2, ws-lines-on-page, output-area |

شكل ( ٨ - ١١ ) صفحة ٧

| | | | |
|---|---|---|---|
| Designer L. Newcomer | System/Program Name PAYROLL | | Date 10/83 |
| Module No. 210 | Module Name WRITE-TO-DISK | | |

| Input | Process | Output |
|---|---|---|
| time card | 1. move all fields from time card to disk record<br>2. write disk record | disk record |

شكل ( ٨ - ١١ ) صفحة ٨

| | | | |
|---|---|---|---|
| Designer L. Newcomer | System/Program Name PAYROLL | | Date 10/83 |
| Module No. 250 | Module Name GET-TIME-CARD | | |

| Input | Process | Output |
|---|---|---|
| time-card | 1. read time-card-file<br>at end: move "yes" to end-time-file-sw | ws-end-time-file-sw |

شكل (٨ - ١١) صفحة ٩

```
       IDENTIFICATION DIVISION.

       PROGRAM-ID.  TIMEEDIT.

       AUTHOR.  LARRY NEWCOMER.
       INSTALLATION.  PENN STATE UNIVERSITY -- YORK CAMPUS.

       ENVIRONMENT DIVISION.

       CONFIGURATION SECTION.
       SOURCE-COMPUTER.  IBM-3081 WITH DEBUGGING MODE.
       OBJECT-COMPUTER.  IBM-3081.

       INPUT-OUTPUT SECTION.
       FILE-CONTROL.

           SELECT TIME-CARD-FILE
               ASSIGN TO TIMECARD
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
           .
           SELECT VALID-TIME-FILE
               ASSIGN TO DISKTIME
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
           .
           SELECT ERROR-LISTING-FILE
               ASSIGN TO ERRORLOG
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL


       DATA DIVISION.

       FILE SECTION.

       FD  TIME-CARD-FILE
           RECORD CONTAINS 80 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .

       01  TIME-CARD-RECORD.
           05  TIME-CARD-ID              PIC X(5).
           05  TIME-CARD-HOURS           PIC 9(2)V9.
           05  TIME-CARD-X-HOURS         REDEFINES TIME-CARD-HOURS
                                         PIC X(3).
           05  TIME-CARD-CLOSING-DATE    PIC X(6).
           05  TIME-CARD-DEPARTMENT      PIC X(4).
           05  FILLER                    PIC X(62).

       FD  VALID-TIME-FILE
           BLOCK CONTAINS 0 RECORDS
           RECORD CONTAINS 23 CHARACTERS
           LABEL RECORDS ARE STANDARD
           .

       01  VALID-TIME-RECORD.
           05  VALID-TIME-ID             PIC X(5).
           05  VALID-TIME-HOURS          PIC S9(2)V9     COMP-3.
           05  VALID-TIME-CLOSING-DATE   PIC X(6).
```

شكل ( ٨ - ١٢ )

```
                05  VALID-TIME-DEPARTMENT    PIC X(4).
                05  VALID-TIME-EDIT-DATE     PIC X(6).

            FD  ERROR-LISTING-FILE
                RECORD CONTAINS 132 CHARACTERS
                LABEL RECORDS ARE OMITTED
                .

            01  ERROR-REPORT-LINE            PIC X(132).

            WORKING-STORAGE SECTION.

            01  PROGRAM-SWITCHES.
                05  WS-END-TIME-FILE-SW      PIC X(3).
                    88  NO-MORE-RECORDS      VALUE "YES".
                    88  RECORD-AVAILABLE     VALUE "NO".
                05  WS-ERROR-DETECTED-SW     PIC X(3).
                    88  VALID-RECORD         VALUE "NO ".
                    88  INVALID-RECORD       VALUE "YES".

            01  PROGRAM-COUNTERS.
                05  WS-PAGE-NUMBER           PIC S9(3)      COMP-3.
                05  WS-NUMBER-EMPLOYEES      PIC S9(5)      COMP-3.
                05  WS-NUMBER-TO-SKIP        PIC S9         COMP SYNC.
                05  WS-LINES-ON-PAGE         PIC S9(2)      COMP SYNC.

            01  PROGRAM-TOTAL-AREAS.
                05  WS-TOTAL-HOURS           PIC S9(7)V9    COMP-3.

           01  PROGRAM-CONSTANTS.
                05  WS-PAGE-SIZE             PIC S9(2)      COMP SYNC
                                             VALUE +50.

            01  WS-HEADING-LINE-1.
                05  FILLER                   PIC X(5)       VALUE SPACES.
                05  FILLER                   PIC X(15)
                                             VALUE "TIME CARD EDIT".
                05  WS-HEADING-MM            PIC Z9.
                05  FILLER                   PIC X          VALUE "/".
                05  WS-HEADING-DD            PIC 99.
                05  FILLER                   PIC X          VALUE "/".
                05  WS-HEADING-YY            PIC 99.
                05  FILLER                   PIC X(3)       VALUE SPACES.
                05  FILLER                   PIC X(5)       VALUE "PAGE ".
                05  WS-HEADING-PAGE          PIC ZZ9.
                05  FILLER                   PIC X(93)      VALUE SPACES.

            01  WS-HEADING-LINE-2.
                05  FILLER                   PIC X(7)       VALUE "  ID".
                05  FILLER                   PIC X(5)       VALUE "HRS".
                05  FILLER                   PIC X(8)       VALUE " DATE".
                05  FILLER                   PIC X(112)     VALUE "DEPT".

            01  WS-DETAIL-LINE.
                05  WS-DETAIL-ID             PIC X(5).
                05  FILLER                   PIC X(2)       VALUE SPACES.
```

شكل ( ٨ - ١٢ ) تكملة

```
          05  WS-DETAIL-HOURS          PIC X(3).
          05  FILLER                   PIC X(2)      VALUE SPACES.
          05  WS-DETAIL-DATE           PIC X(6).
          05  FILLER                   PIC X(2)      VALUE SPACES.
          05  WS-DETAIL-DEPARTMENT     PIC X(4).
          05  FILLER                   PIC X(108)    VALUE SPACES.

      01  WS-UNDER-LINE.
          05  WS-UNDER-ID              PIC X(5).
          05  FILLER                   PIC X(2)      VALUE SPACES.
          05  WS-UNDER-HOURS           PIC X(3).
          05  FILLER                   PIC X(2)      VALUE SPACES.
          05  WS-UNDER-DATE            PIC X(6).
          05  FILLER                   PIC X(2)      VALUE SPACES.
          05  WS-UNDER-DEPARTMENT      PIC X(4).
          05  FILLER                   PIC X(108)    VALUE SPACES.

      PROCEDURE DIVISION.

      000-EDIT-TIME-CARDS.

          PERFORM 100-INITIALIZE
          PERFORM 200-PRODUCE-EDIT-LISTING
              UNTIL NO-MORE-RECORDS
          PERFORM 300-TERMINATE
          STOP RUN


      100-INITIALIZE.

     D    DISPLAY "100 ENTERED"
          .
      200-PRODUCE-EDIT-LISTING.

     D    DISPLAY "200 ENTERED"
     D    MOVE "YES" TO WS-END-TIME-FILE-SW
          .

      300-TERMINATE.

     D    DISPLAY "300 ENTERED"
          .
```

شكل ( ٨ - ١٢ ) تكملة

```
        PROCEDURE DIVISION.

        000-EDIT-TIME-CARDS.

            PERFORM 100-INITIALIZE
            PERFORM 200-PRODUCE-EDIT-LISTING
                UNTIL NO-MORE-RECORDS
            PERFORM 300-TERMINATE
            STOP RUN
            .

        100-INITIALIZE.

      D     DISPLAY "100 ENTERED"
            OPEN    INPUT   TIME-CARD-FILE
                    OUTPUT  VALID-TIME-FILE
                            ERROR-LISTING-FILE
            MOVE "NO" TO                WS-END-TIME-FILE-SW
            MOVE ZERO TO                WS-PAGE-NUMBER
                                        WS-NUMBER-EMPLOYEES
                                        WS-TOTAL-HOURS
            MOVE WS-PAGE-SIZE TO        WS-LINES-ON-PAGE
            ACCEPT WS-SYSTEM-DATE FROM  DATE
            MOVE SYSTEM-YY TO           WS-HEADING-YY
            MOVE SYSTEM-MM TO           WS-HEADING-MM
            MOVE SYSTEM-DD TO           WS-HEADING-DD
            .

        200-PRODUCE-EDIT-LISTING.

      D     DISPLAY "200 ENTERED"
            PERFORM 250-GET-TIME-CARD
            IF RECORD-AVAILABLE
                PERFORM 240-VALIDATE-FIELDS
                IF VALID-RECORD
                    PERFORM 210-WRITE-TO-DISK
                ELSE
                    PERFORM 220-PRODUCE-ERROR-LISTING
            .

        210-WRITE-TO-DISK.

      D     DISPLAY "210 ENTERED"
            .

        220-PRODUCE-ERROR-LISTING.

      D     DISPLAY "220 ENTERED"
            .

        230-PRINT-OUTPUT-AREA.

      D     DISPLAY "230 ENTERED"
            .

        240-VALIDATE-FIELDS.

      D     DISPLAY "240 ENTERED"
      D     ON 1 AND EVERY 2
```

شكل ( ٨ - ١٣ )

```
D          MOVE "YES" TO WS-ERROR-DETECTED-SW
D      ELSE
D          MOVE "NO" TO WS-ERROR-DETECTED-SW


 250-GET-TIME-CARD.

     READ TIME-CARD-FILE
         AT END
             MOVE "YES" TO WS-END-TIME-FILE-SW


 300-TERMINATE.

D    DISPLAY "300 ENTERED"
     MOVE WS-NUMBER-EMPLOYEES     TO WS-FOOTING-COUNT
     MOVE WS-TOTAL-HOURS          TO WS-FOOTING-TOTAL
     MOVE WS-FOOTING-LINE         TO WS-OUTPUT-AREA
     MOVE WS-SKIP-BEFORE-FOOTING  TO WS-NUMBER-TO-SKIP
     MOVE WS-FOOTING-LINE         TO WS-OUTPUT-AREA
     PERFORM 230-PRINT-OUTPUT-AREA

     CLOSE   TIME-CARD-FILE
             VALID-TIME-FILE
             ERROR-LISTING-FILE

```

شكل ( ٨ - ١٣ )

```
 IDENTIFICATION DIVISION.

 PROGRAM-ID.   TIMEEDIT.

 AUTHOR.   LARRY NEWCOMER.
 INSTALLATION.   PENN STATE UNIVERSITY -- YORK CAMPUS.

 ENVIRONMENT DIVISION.

 CONFIGURATION SECTION.
 SOURCE-COMPUTER.   IBM-3081 WITH DEBUGGING MODE.
 OBJECT-COMPUTER.   IBM-3081.

 INPUT-OUTPUT SECTION.
 FILE-CONTROL.

     SELECT TIME-CARD-FILE
         ASSIGN TO TIMECARD
         ORGANIZATION IS SEQUENTIAL
         ACCESS IS SEQUENTIAL
     .
     SELECT VALID-TIME-FILE
         ASSIGN TO DISKTIME
         ORGANIZATION IS SEQUENTIAL
```

شكل ( ٨ - ١٤ )

```
               ACCESS IS SEQUENTIAL
           .
           SELECT ERROR-LISTING-FILE
               ASSIGN TO ERRORLOG
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL


       DATA DIVISION.

       FILE SECTION.

       FD  TIME-CARD-FILE
           RECORD CONTAINS 80 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .

       01  TIME-CARD-RECORD.
           05  TIME-CARD-ID             PIC X(5).
           05  TIME-CARD-HOURS          PIC 9(2)V9.
           05  TIME-CARD-X-HOURS        REDEFINES TIME-CARD-HOURS
                                        PIC X(3).
           05  TIME-CARD-CLOSING-DATE   PIC X(6).
           05  TIME-CARD-DEPARTMENT     PIC X(4).
           05  FILLER                   PIC X(62).

       FD  VALID-TIME-FILE
           BLOCK CONTAINS 0 RECORDS
           RECORD CONTAINS 23 CHARACTERS
           LABEL RECORDS ARE STANDARD
           .

       01  VALID-TIME-RECORD.
           05  VALID-TIME-ID            PIC X(5).
           05  VALID-TIME-HOURS         PIC S9(2)V9    COMP-3.
           05  VALID-TIME-CLOSING-DATE  PIC X(6).
           05  VALID-TIME-DEPARTMENT    PIC X(4).
           05  VALID-TIME-EDIT-DATE     PIC X(6).

       FD  ERROR-LISTING-FILE
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED


       01  ERROR-REPORT-LINE            PIC X(132).

       WORKING-STORAGE SECTION.

       01  PROGRAM-SWITCHES.
           05  WS-END-TIME-FILE-SW      PIC X(3).
               88  NO-MORE-RECORDS      VALUE "YES".
               88  RECORD-AVAILABLE     VALUE "NO".
           05  WS-ERROR-DETECTED-SW     PIC X(3).
               88  VALID-RECORD         VALUE "NO ".
               88  INVALID-RECORD       VALUE "YES".
```

شكل (٨ - ١٤) تكملة

```

       01  PROGRAM-COUNTERS.
           05  WS-PAGE-NUMBER          PIC S9(3)      COMP-3.
           05  WS-NUMBER-EMPLOYEES     PIC S9(5)      COMP-3.
           05  WS-NUMBER-TO-SKIP       PIC S9         COMP SYNC.
           05  WS-LINES-ON-PAGE        PIC S9(2)      COMP SYNC.

       01  PROGRAM-TOTAL-AREAS.
           05  WS-TOTAL-HOURS          PIC S9(7)V9    COMP-3.

       01  PROGRAM-CONSTANTS.
           05  WS-PAGE-SIZE            PIC S9(2)      COMP SYNC
                                       VALUE +50.
           05  WS-SKIP-BEFORE-HEADING  PIC S9(2)      COMP SYNC
                                       VALUE +2.
           05  WS-SKIP-BEFORE-FOOTING  PIC S9(2)      COMP SYNC
                                       VALUE +4.
           05  WS-SKIP-BEFORE-DETAIL   PIC S9(2)      COMP SYNC
                                       VALUE +2.
           05  WS-SKIP-BEFORE-UNDER    PIC S9(2)      COMP SYNC
                                       VALUE +1.
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
       PROCEDURE DIVISION.

       000-EDIT-TIME-CARDS.

           PERFORM 100-INITIALIZE
           PERFORM 200-PRODUCE-EDIT-LISTING
               UNTIL NO-MORE-RECORDS
           PERFORM 300-TERMINATE
           STOP RUN


       100-INITIALIZE.

      D    DISPLAY "100 ENTERED"
           OPEN    INPUT   TIME-CARD-FILE
                   OUTPUT  VALID-TIME-FILE
                           ERROR-LISTING-FILE
           MOVE "NO" TO            WS-END-TIME-FILE-SW
           MOVE ZERO TO            WS-PAGE-NUMBER
                                   WS-NUMBER-EMPLOYEES
                                   WS-TOTAL-HOURS
           MOVE WS-PAGE-SIZE TO    WS-LINES-ON-PAGE
           ACCEPT WS-SYSTEM-DATE FROM DATE
           MOVE SYSTEM-YY TO       WS-HEADING-YY
           MOVE SYSTEM-MM TO       WS-HEADING-MM
           MOVE SYSTEM-DD TO       WS-HEADING-DD


       200-PRODUCE-EDIT-LISTING.

      D    DISPLAY "200 ENTERED"
           PERFORM 250-GET-TIME-CARD
           IF RECORD-AVAILABLE
               PERFORM 240-VALIDATE-FIELDS
               IF VALID-RECORD
                   ADD 1               TO WS-NUMBER-EMPLOYEES
                   ADD TIME-CARD-HOURS TO WS-TOTAL-HOURS
```

شكل (٨ - ١٤) تكملة

```
                      PERFORM 210-WRITE-TO-DISK
                  ELSE
                      PERFORM 220-PRODUCE-ERROR-LISTING


       210-WRITE-TO-DISK.

      D    DISPLAY "210 ENTERED"
           MOVE TIME-CARD-ID            TO VALID-TIME-ID
           MOVE TIME-CARD-HOURS         TO VALID-TIME-HOURS
           MOVE TIME-CARD-CLOSING-DATE  TO VALID-TIME-CLOSING-DATE
           MOVE TIME-CARD-DEPARTMENT    TO VALID-TIME-DEPARTMENT
           MOVE WS-SYSTEM-DATE          TO VALID-TIME-EDIT-DATE
           WRITE VALID-TIME-RECORD


       220-PRODUCE-ERROR-LISTING.

      D    DISPLAY "220 ENTERED"
           MOVE TIME-CARD-ID            TO WS-DETAIL-ID
           MOVE TIME-CARD-X-HOURS       TO WS-DETAIL-HOURS
           MOVE TIME-CARD-CLOSING-DATE  TO WS-DETAIL-DATE
           MOVE TIME-CARD-DEPARTMENT    TO WS-DETAIL-DEPARTMENT

           MOVE WS-DETAIL-LINE          TO WS-OUTPUT-AREA
           MOVE WS-SKIP-BEFORE-DETAIL   TO WS-NUMBER-TO-SKIP
           PERFORM 230-PRINT-OUTPUT-AREA

           MOVE WS-UNDER-LINE           TO WS-OUTPUT-AREA
           MOVE WS-SKIP-BEFORE-UNDER    TO WS-NUMBER-TO-SKIP
           PERFORM 230-PRINT-OUTPUT-AREA


       230-PRINT-OUTPUT-AREA.

      D    DISPLAY "230 ENTERED"
           IF WS-LINES-ON-PAGE + WS-NUMBER-TO-SKIP
               GREATER THAN WS-PAGE-SIZE
                   ADD 1 TO WS-PAGE-NUMBER
                   MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE
                   WRITE ERROR-REPORT-LINE
                       FROM WS-HEADING-LINE-1
                       AFTER ADVANCING PAGE
                   WRITE ERROR-REPORT-LINE
                       FROM WS-HEADING-LINE-2
                       AFTER ADVANCING WS-SKIP-BEFORE-HEADING LINES
                   MOVE WS-SKIP-BEFORE-HEADING TO WS-LINES-ON-PAGE
           .
           WRITE ERROR-REPORT-LINE
               FROM WS-OUTPUT-AREA
               AFTER ADVANCING WS-NUMBER-TO-SKIP LINES
           ADD WS-NUMBER-TO-SKIP TO WS-LINES-ON-PAGE
           .

       240-VALIDATE-FIELDS.

      D    DISPLAY "240 ENTERED"
           MOVE "NO"   TO WS-ERROR-DETECTED-SW
           MOVE SPACES TO WS-UNDER-LINE
           IF TIME-CARD-ID NOT NUMERIC
               MOVE ALL "-" TO WS-UNDER-ID
               MOVE "YES"   TO WS-ERROR-DETECTED-SW
```

شكل (٨ - ١٤) تكملة

```
                   .
                   IF TIME-CARD-HOURS NOT NUMERIC
                       MOVE ALL "-" TO WS-UNDER-HOURS
                       MOVE "YES"   TO WS-ERROR-DETECTED-SW
                   .
                   IF TIME-CARD-CLOSING-DATE NOT NUMERIC
                       MOVE ALL "-" TO WS-UNDER-DATE
                       MOVE "YES"   TO WS-ERROR-DETECTED-SW
                   .
                   IF TIME-CARD-DEPARTMENT NOT NUMERIC
                       MOVE ALL "-" TO WS-UNDER-DEPARTMENT
                       MOVE "YES"   TO WS-ERROR-DETECTED-SW
                   .

               250-GET-TIME-CARD.

                   READ TIME-CARD-FILE
                       AT END
                           MOVE "YES" TO WS-END-TIME-FILE-SW


               300-TERMINATE.

              D    DISPLAY "300 ENTERED"
                   MOVE WS-NUMBER-EMPLOYEES    TO WS-FOOTING-COUNT
                   MOVE WS-TOTAL-HOURS         TO WS-FOOTING-TOTAL
                   MOVE WS-FOOTING-LINE        TO WS-OUTPUT-AREA
                   MOVE WS-SKIP-BEFORE-FOOTING TO WS-NUMBER-TO-SKIP
                   MOVE WS-FOOTING-LINE        TO WS-OUTPUT-AREA
                   PERFORM 230-PRINT-OUTPUT-AREA

                   CLOSE   TIME-CARD-FILE
                           VALID-TIME-FILE
                           ERROR-LISTING-FILE

```

```
     TIME CARD EDIT  5/09/83   PAGE    1
  ID   HRS   DATE   DEPT
1 111  111  111111  1111
-----
11 11  111  111111  1111
-----
111 1  111  111111  1111
-----
11111   11  111111  1111
       ---
11111  1 1  111111  1111
       ---
.
.
.
11111  111  111111  11 1
                    ----
     TIME CARD EDIT  5/09/83   PAGE    2
  ID   HRS   DATE   DEPT
11111  111  111111  111
                    ----

-----  ---  ------  ----

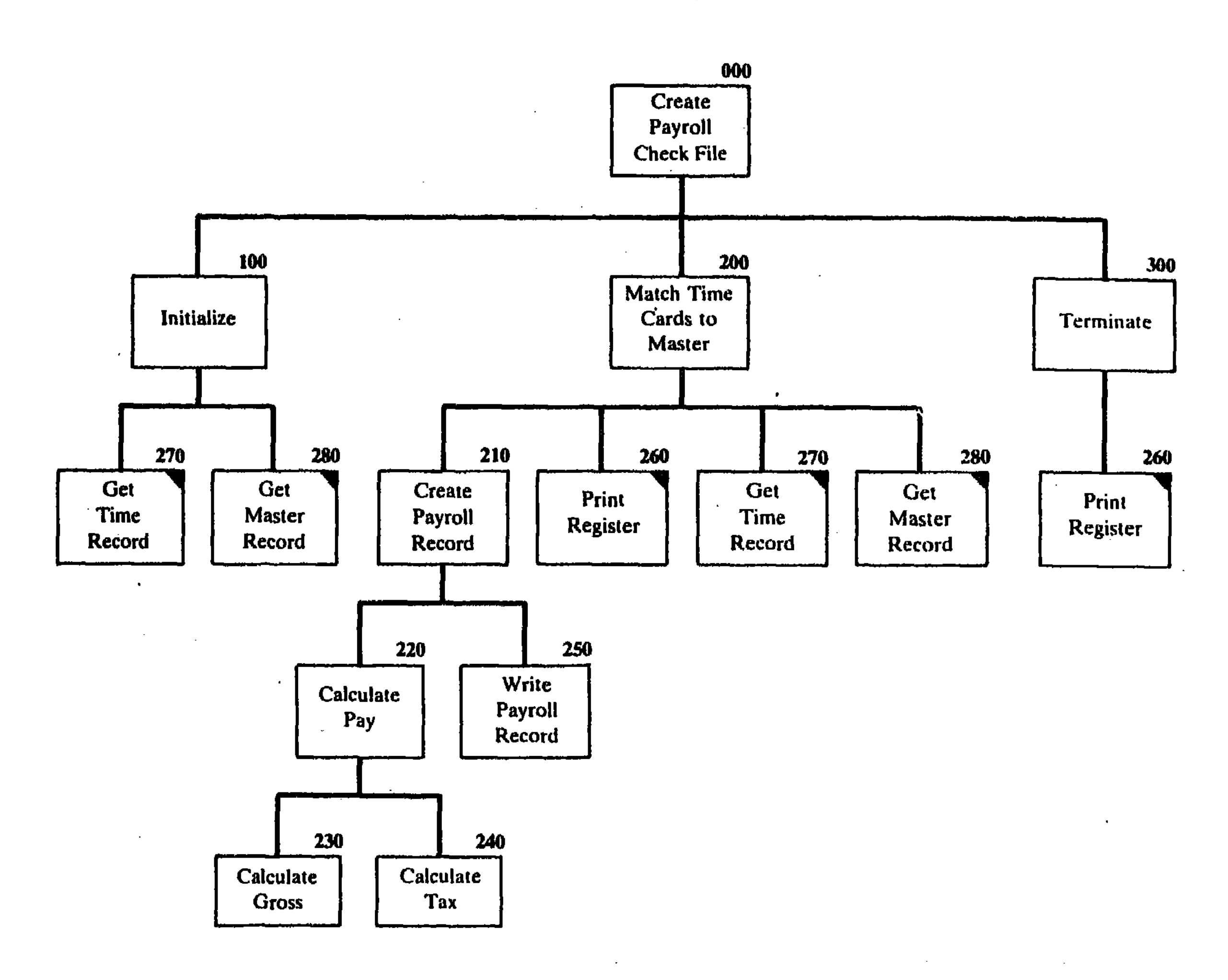
# EMPLOYEES=     2   TOTAL HOURS=       33.3
```

شكل (٨ - ١٤) تكملة

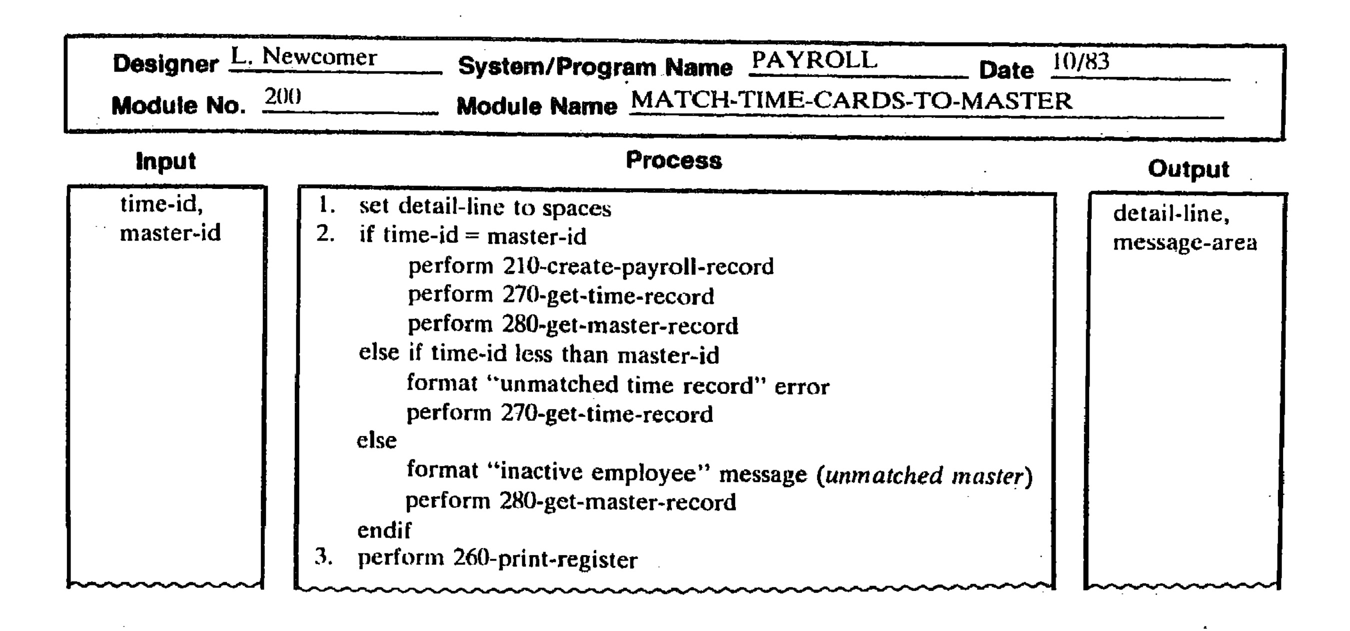
المسأله رقم ٢٢ من الفصل الحادى عشر تعزز الحل السابق .

٨ – ٣١ ارسم خريطة هيكل لبرنامج رواتب المسألة السابقة .

انظر شكل ٨ – ١٥. قلب الخوارزمى هو الجزء الذى يجعل السجل المنطقى لبطاقه وقت صحيحة ( من ملف قرص يتم إنتاجه بواسطة برنامج المسأله السابقه ) متفقا مع السجلات المنطقية المناظرة من ملف العاملين الرئيسى ، والذى يعطى معدل الأجر لكل عامل . بافتراض أن كلاً من الملفين مرتب ترتيبا تصاعديا ، طبقا لرقم تعريف العامل ، فيمكن عمل خريطة هيبو لهذا الجزء كما تبدو فى شكل ٨ – ١٦ ( انظر المسألة رقم ٢٣ فى الفصل الحادى عشر لرؤية برنامج الكوبل كاملا . )




شكل ( ٨ – ١٥ )

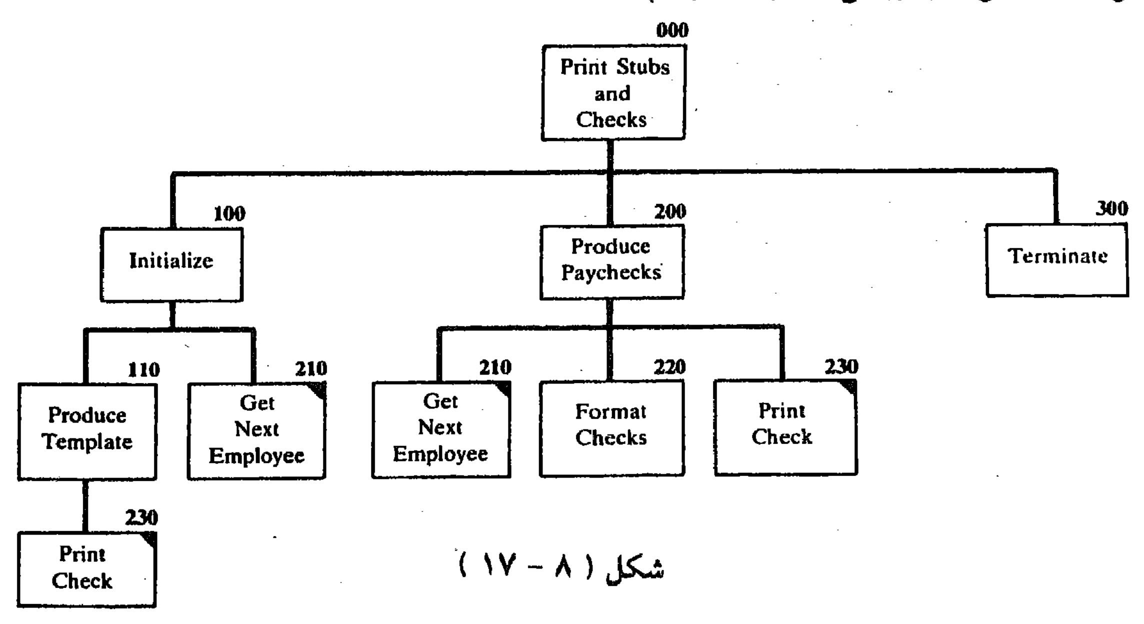

Designer L. Newcomer System/Program Name PAYROLL Date 10/83
Module No. 200 Module Name MATCH-TIME-CARDS-TO-MASTER

| Input | Process | Output |
|---|---|---|
| time-id,<br>master-id | 1. set detail-line to spaces<br>2. if time-id = master-id<br>perform 210-create-payroll-record<br>perform 270-get-time-record<br>perform 280-get-master-record<br>else if time-id less than master-id<br>format "unmatched time record" error<br>perform 270-get-time-record<br>else<br>format "inactive employee" message (*unmatched master*)<br>perform 280-get-master-record<br>endif<br>3. perform 260-print-register | detail-line,<br>message-area |

شكل ( ٨ - ١٦ )

٨ - ٣٢ ارسم خريطة هيكل واختبر طباعة البرنامج لنظام رواتب المسألتين ٣٠ و ٣١ السابقتين .

انظر شكل ٨ - ١٧ ... يمكن استخدام عبارة ACCEPT فى مقطع الكوبل 100 - INITIALIZE ، وذلك للحصول على تاريخ اليوم ، والتاكد من الرقم الذى سيطبع على أول شيك ، وتنتج أرقام الشيكات التالية باضافة 1 لكل شيك . ( يعطى مثال ١٠ فى الفصل الثانى عشر البرنامج كاملا بلغة الكوبل . )




شكل ( ٨ - ١٧ )

٨ - ٣٣ بالإشاره إلى المسأله رقم ٣٠ من هذا الفصل .. اكتب شفرة شبيهة لاختبار متتال sequence check يؤكد أن سجلات بطاقات العمل موجودة في ترتيب تصاعدى ؛ طبقا لرقم تعريف العامل ( وعلى هذا .. يمكن أن تتفق بطريقة مناسبة مع سجلات الملف الرئيسى للعاملين ، كما هو مطلوب فى المسألة رقم ٣١ من هذا الفصل ) .

```
set end-of-file "off"
set previous-control-field to low-values
read input-file
     at end:   set end-of-file "on"

perform until end-of-file is "on"
     if current-control-field greater than previous-control-field

          (process an in-sequence record)

          set previous-control-field to current-control-field
     else
          (current logical record is out of sequence)

          perform record-out-of-sequence-routine
     endif
     read input-file
          at end:   set end-of-file "on"
endperform
```

لاحظ أن وضع قيمة ابتدائية low - value لحقل التحكم السابق .. تضمن أن حقل التحكم السابق أقل من حقل التحكم الحالى لأول سجل منطقى . لاحظ كذلك انه فى كل مرة يحدث تشغيل لسجل منطقى فى الترتيب ؛ فيوضع عنصر بيانات حقل التحكم السابق ، مساويا لمحتويات حقل التحكم الحالى . وهذا يعد حقل التحكم السابق للمقارنة مع السجل المنطقى التالى فى الملف .

# الفصل التاسع

# التصحيح

# Debugging

يشير التصحيح debugging إلى عملية تعريف أسباب أخطاء البرنامج وتصحيح هذه الأخطاء . وهناك ثلاثة أنواع لاخطاء البرنامج ، يجب التعامل معها ، هى :

● أخطاء تكوينية : أخطاء فى القواعد عند استخدام لغة البرمجة ( الكوبل) ، وتكتشف بواسطة المترجم عند محاولته الترجمة من لغة الكوبل الى لغة الآلة .

● أخطاء منطقية جسيمة : تتسبب فى إنهاء البرنامج نهاية غير طبيعية ، وتكتشف بواسطة نظم المكونات ، أو نظام التشغيل أثناء تنفيذ البرنامج ببيانات اختبارية (أى بعد تصحيح كل الأخطاء التكوينية) ، عندما يصبح تنفيذ تعليمات لغة الآلة ، أو تصبح خدمة نظام التشغيل غير صحيحة .

● أخطاء منطقية غير جسيمة : والتى ينفذ فيها البرنامج كلية حتى عبارة STOP RUN ، ولكنه لا يعطى النتيجة الصحيحة يجب أن تزال كل الأخطاء التكوينية والأخطاء المنطقية الجسيمة من البرنامج ، قبل أن تعطى الأخطاء المنطقية غير الجسيمة فرصة لإظهار نفسها .

## ٩ - ١ تصحيح الاخطاء التكوينية

تعد الأخطاء التكوينية أسهل فى تصحيحها ، لأن مترجم الكوبل يطبع رسائل تشخيصية للخطأ ، تعرف الأخطاء ، وعبارة الكوبل التى حدثت فيها هذه الأخطاء . عادة ما تظهر هذه الرسائل فى نهاية قائمة برنامج المصدر ، ويفحص كل خطأ لتصحح العبارة التى تحتويه بما يتفق مع قواعد الكوبل .

**مثال ٩ - ١ :**

شكل ٩ - ١ ، هو قائمة برنامج مصدر بكوبل IBM OS/VS ، يبين الأخطاء التكوينية مع رسائل تشخيص الخطأ . تحدد كل رسالة خطأ رقم ينتجه الكمبيوتر لبطاقة العبارة الموجود بها الخطأ ، ورمز يعرف الخطأ ودرجة خطورته ، وعبارة باللغة الإنجليزية تحاول وصف الخطأ . وفيما يلى مناقشه لأخطاء تكوينية مختارة .

```
           IDENTIFICATION DIVISION.

           PROGRAM-ID.  SYNTAX.

           AUTHOR.  LARRY NEWCOMER.
           INSTALLATION.  PENN STATE UNIVERSITY -- YORK CAMPUS.

           ENVIRONMENT DIVISION.

           CONFIGURATION SECTION.
           SOURCE-COMPUTER.  IBM-3081
           OBJECT-COMPUTER.  IBM-3081.

           INPUT-OUTPUT SECTION.
           FILE-CONTROL.

               SELECT TIME-CORD-FILE
                   ASSIGN TO TIMECARD
                   ORGANIZATION IS SEQUENTIAL
                   ACCESS IS SEQUENTIAL
               .
               SELECT VALID-TIME-FILE
                   ASSIGN TO DISKTIME
                   ORGANIZATION IS SEQUENTIAL
                   ACCESS IS SEQUENTIAL
               .
               SELECT ERROR-LISTING-FILE
                   ASSIGN TO ERRORLOG
                   ORGANIZATION IS SEQUENTIAL
                   ACCESS IS SEQUENTIAL
               .

           DATA DIVISION.

           FILE SECTION.

           FD  TIME-CARD-FILE
               RECORD CONTAINS 80 CHARACTERS
               LABEL RECORDS ARE OMITTED
               .

           01  TIME-CARD-RECORD.
               05  TIME-CARD-ID               PIC S(5).
               05  TIME-CARD-HOURS            PIC 9(2)V9.
               05  TIME-CARD-X-HOURS          REDFINES TIME-CARD-HOURS
                                              PIC X(3).
               05  TIME-CARD-CLOSING-DATE     PIC X(6).
               05  TIME-CARD-DEPARTMENT       PIC X(4).
               05  FILLER                     PIC X(62).

           FD  VALID-TIME-FILE
               BLOCK CONTAINS 0 RECORDS
               RECORD CONTAINS 23 CHARACTERS
               LABEL RECORDS ARE STANDARD

           01  VALID-TIME-RECORD.
               05  VALID-TIME-ID              PIC X(5).
               05  VALID-TIME-HOURS           PIC S9(2)V9     COMP-3.
               05  VALID-TIME-CLOSING-DATE    PIC X(6).
               05  VALID-TIME-DEPARTMENT      PIC X(4).
               05  VALID-TIME-EDIT-DATE       PIC X(6).

           FD  ERROR-LISTING-FILE
               RECORD CONTAINS 132 CHARACTERS
               LABEL RECORDS ARE OMITTED
               .

           01  ERROR-REPORT-LINE              PIC X(132).

           WORKING-STORAGE SECTION.

.
.
.
           01  WS-HEADING-LINE-1.
               05  FILLER                     PIC X(5)        VALUE SPACES.
               05  FILLER                     PIC X(15)
                                              VALUE "TIME CARD EDIT".
               05  WS-HEADING-MM              PIC Z9.
               05  FILLER                     PIC X           VALUE "//".
               05  WS-HEADING-DD              PIC 99.
               05  FILLER                     PIC X           VALUE "/".
               05  WS-HEADING-YY              PIC 99.
               05  FILLER                     PIC X(3)        VALUE SPACES.
               05  FILLER                     PIC X(5)        VALUE "PAGE ".
               05  WS-HEADING-PAGE            PIC ZZ9.
               05  FILLER                     PIC X(93)       VALUE SPACES.
```

شكل ( ٩ - ١ )

```
.
.
.
00135
00136          01  'S-UNDER-LINE.
00137              05  WS-UNDER-ID              PIC X(5).
00138              05  FILLER                   PIC X(2)       VALUE SPICES.
00139              05  WS-UNDER-HOURS           PIC X(3).
00140              05  FILLER                   PIC X(2)       VALUE SPACES.
00141              05  WS-UNDER-DATE            PIC X(6).
00142              05  FILLER                   PIC X(2)       VALUE SPACES.
00143              05  WS-UNDER-DEPARTMENT      PIC X(4).
00144              05  FILLER                   PIC X(108)     VALUE SPACES.
.
.
.
00158          PROCEDURE DIVISION.
00159
00160          000-EDIT-TIME-CARDS.
00161
00162              PERFORM 100-INITIALIZE
00163              PERFORM 200-PRODUCE-EDIT-LISTING
00164                  UNTIL NO-MORE-RECORDS
00165              PERFORM 300-TERMINATE
00166              STOP RUN
00167              .
00168
00169          100-INITIALIZE.
00170
00171        D     DISPLAY "100 ENTERED"
00172              OPEN    INPUT   TIME-CARD-FILE
00173                      OUTPUT  VALID-TIME-FILE
00174                              ERROR-LISTING-FILE
00175              MOVE "NO" TO                 WS-END-TIME-FILE-SW
00176              MOVE ZERO TO                 WS-PAGE-NUMBER
00177                                           WS-NUMBER-EMPLOYEES
00178                                           WS-TOTAL-HOURS
00179              MOVE WS-PAGE-SIZE TO         WS-LINES-ON-PAGE
00180              ACCEPT WS-SYSTEM-DATE FROM DATE
00181              MOVE SYSTEM-YY TO            WS-HEADING-YY
00182              MOVE SYSTEM-MM TO            WS-HEADING-MM
00183              MOVE SYSTEM-DD TO            WS-HEADING-DD
00184
00185
00186          200-PRODUCE-EDIT-LISTING.
00187
00188        D     DISPLAY "200 ENTERED"
00189              PERFORM 250-GET-TIME-CARD
00190              IF RECORD-AVAILABLE
00191                  PERFORM 240 VALIDATE-FIELDS
00192                  IF VALID-RECORD EQUAL "YES"
00193                      ADD 1                  TO WS-NUMBER-EMPLOYEES
00194                      ADD TIME-CARD-HOURS TO WS-TOTAL-HOURS
00195                      PERFORM 210-WRITE-TO-DISK
00196                  ELSE
00197                      PERFORM 220-PRODUCE-ERROR-LISTING
00198              .
00199
00200          210-WRITE-TO-DISK.
00201
00202        D     DISPLAY "210 ENTERED"
00203              MOVE TIME-CARD-ID            TO VALID-TIME-ID
00204              MOVE TIME-CARD-HOURS         TO VALID-TIME-HOURS
00205              MOVE TIME-CARD-CLOSING-DATE  TO VALID-TIME-CLOSING-DATE
00206              MOVE TIME-CARD-DEPARTMENT    TO VALID-TIME-DEPARTMENT
00207              MOVE WS-SYSTEM-DATE          TO VALID-TIME-EDIT-DATE
00208              WRITE VALID-TIME-FILE
00209              .
.
.
.
CARD   ERROR MESSAGE

12     IKF1043I-W    END OF SENTENCE SHOULD PRECEDE OBJECT-COMPUTER . ASSUMED PRESENT.
40     IKF1056I-E    FILE-NAME NOT DEFINED IN A SELECT. DESCRIPTION IGNORED.
56     IKF1043I-W    END OF SENTENCE SHOULD PRECEDE 01 . ASSUMED PRESENT.
111    IKF2126I-C    VALUE CLAUSE LITERAL TOO LONG. TRUNCATED TO PICTURE SIZE.
138    IKF1017I-E    SPICES INVALID IN VALUE CLAUSE. SKIPPING TO NEXT CLAUSE.
172    IKF3001I-E    TIME-CARD-FILE NOT DEFINED. DELETING TILL LEGAL ELEMENT FOUND.
191    IKF3001I-E    240 NOT DEFINED. STATEMENT DISCARDED.
191    IKF3001I-E    VALIDATE-FIELDS NOT DEFINED.
192    IKF4030I-E    FOUND ' EQUAL ' AFTER CONDITION. EXPECT 'OR', 'AND' OR VERB TO IMMEDIATELY FOLLOW
                     CONDITION. DELETING TILL ONE OF THESE IS FOUND.
194    IKF3001I-E    TIME-CARD-HOURS NOT DEFINED. SUBSTITUTING TALLY .
203    IKF3001I-E    TIME-CARD-ID NOT DEFINED. DISCARDED.
204    IKF3001I-E    TIME-CARD-HOURS NOT DEFINED. DISCARDED.
205    IKF3001I-E    TIME-CARD-CLOSING-DATE NOT DEFINED. DISCARDED.
206    IKF3001I-E    TIME-CARD-DEPARTMENT NOT DEFINED. DISCARDED.
```

شكل ( ٩ - ١ ) تكملة

• **البطاقة** 12 : فى كوبل IBM OS / VS ، رمز الرساله الذى ينتهى بالحرف W (مثل IKF 10431 - W ) يحدد تحذيرا Warning ، وهذا يعنى أن الخطأ بسيط ، وأن المترجم كان قادرا على تصحيحه . فى هذه الحالة ، نسى المبرمج النقطة التى تنهى مقطع كمبيوتر المصدر ، وقد افترض المترجم أن النقطة موجودة ، على أية حال .. يستطيع المترجم ترجمة العبارات التى تنتج رسائل تحذيرية .

• **البطاقة** 111 : فى كوبل IMB OS/VS ، يحدد رمز الرسالة الذى ينتهى بالحرف C ( مثل IKF 2126I - C ) أخطاء شرطية conditional errors . هذه الأخطاء أكثر شدة من الأخطاء التحذيرية ، إلا أن المترجم لايزال قادرا على تخمين ما يريده المبرمج . ويستخدم هذا التخمين فى ترجمة العبارة ، إلا أن احتمال أن الترجمة هى ما يعنيه المبرمج ليس مرتفعاً كما فى حالة الأخطاء التحذيرية . فى هذه الحالة ، كان جزء VALUE أكبر من حجم PICTURE ، وعلى هذا .. حذف المترجم مازاد عن عدد الخانات ( "//" VALUE أصبحت "/" لتناسب PIC X ) .

• **البطاقة** 192 : فى كوبل IBM OS / VS ، رمز الرسالة الذى ينتهى بالحرف E ( مثل IKF 4030 I - E ) يحدد أخطاء جسيمة severe errors . فى هذه الحالة .. لا يستطيع المترجم حتى أن يخمن ما تعنيه العبارة . وعلى هذا ، لا يمكن ترجمة العبارة إلى لغة الآلة .. و لذلك .. لن يتواجد برنامج هدف لتنفيذه ببيانات اختبارية . لاحظ أنه فى هذه اللحظة لا تذكر رسالة الخطأ الإنجليزية للمبرمج الخطأ الموجود فعلا فى العبارة . والخطأ الفعلى هو أن المبرمج استخدم اسم شرطى - VALID RECORD بدلا من اسم بيانات WS- ERROR - DETECTION -SW فى شرط العلاقة VALID- RECORD EQUAL "YES" .

• **البطاقة** 40 : هذا الخطأ التكوينى يوضح حقيقة أن رسائل الخطأ التشخيصية لا تشير دائما إلى رقم سطر العبارة التى تسببت فعلا فى الخطأ . وهنا .. يعد السطر 40 نقطة مرتبة تنهى TIME - CARD - FILE . وتقول رسالة الخطأ FILE NAME NOT DEFINED IN A SELECT . بالرغم من أن رسالة الخطأ تشير للسطر 40 .. إلا أن المشكلة الفعلية هى فى خطأ كتابة فى السطر 17 ، والذى يقرأ SELECT TIME - CARD - FILE .

• **البطاقات** 206, 205, 204, 203, 194 : بسبب خطأ البطاقة 40 .. لم يتم تشغيل FD ووصف المستوى 01 للملف - TIME CARD - FILE بواسطة المترجم . وعلى هذا .. تشتمل كل إشارات جزء الإجراءات إلى الحقول الموجودة فى - TIME CARD - RECORD على أخطاء منطقية . وتقول رسالة الخطأ إن الأوقات غير معرفة NOT DEFINED .. ومن وجهة نظر المترجم .. فهى بالرغم من تعريف المبرمج لها فى الأسطر من 42 الى 49 فى جزء البيانات غير معرفة للمترجم .

## ٩ - ٣ تصحيح الأخطاء المنطقية الجسيمة

عندما يتسبب خطأ جسيم فى إنهاء البرنامج نهاية غير طبيعية .. يلغى نظام التشغيل تنفيذ بقية البرنامج ، بل أكثر من ذلك أنه إذا قامت لغة تحكم البرنامج الخاصة بالبرنامج بطلبه .. فإن نظام التشغيل يطبع نفايا الفصل termination dump (فى ملف طباعة محدد فى لغة تحكم العمل ) . تقدم نفايا الفصل أحد أو كلا نوعى المعلومات التالية والمفيدة فى تصحيح الأخطاء المنطقية الجسيمة :

نفايا ذاكرة : توضح نفايا الذاكرة محتويات ذاكرة الكمبيوتر فى وقت حدوث الخطأ الجسيم ، ويمكن عادة تحديد سبب الخطأ بدراسة محتويات عناصر بيانات خرجة .

تتبع برنامج : يسجل تتبع البرنامج program trace كل التعليمات المنفذة بواسطة الكمبيوتر قبل وعند وقت إنهاء البرنامج نهاية غير طبيعية . فمثلا .. يمكن أن يطبع رقم السطر الذى ينتجه المترجم لكل عبارة قبل ترجمة لغة الآلة مباشرة للعبارة التى نفذت .

فى بعض نظم الكمبيوتر ، لا تتاح إلا نفايا سادسة عشرية نمطية لنظام التشغيل فقط ، وهذه هى نفايا الذاكرة مطبوعة، كسلسلة من أرقام للأساس 16 . ويجب أن يتعلم مبرمج الكوبل الجاد لغة مجمع assembler language على الأقل ليكون قادرا على تفسير نفايا ذاكرة الكمبيوتر . تقدم بعض أجهزة الكمبيوتر الاخرى نفايا رمزية symbolic dumps حيث تطبع محتويات بعض أو كل عناصر جزء البيانات ، عندما ينتهى البرنامج نهاية غير طبيعية (نفايا ذاكرة) ، وذلك مع تتبع البرنامج، وتطبع عناصر البيانات فى حالة DISPLAY مع تسميتها بأسماء كوبل .

## ٩ - ٣ تصحيح الأخطاء المنطقية غير الجسيمة

بينما تنتج الأخطاء التكوينية رسائل ، وتنتج الأخطاء المنطقية الجسيمة نفايا الإنهاء .. فإن الأخطاء المنطقية غير الجسيمة تكون أكثر صعوبة فى التصحيح ، حيث لا تتحدد عبارة خاصة بأنها سبب الخطأ . ويترك تحديد الخطأ وسببه لبراعة المبرمج .

هناك بعض سمات للغة الكوبل ، صممت خصيصا لمساعدة المبرمج فى تعريف الاخطاء المنطقية غير الجسيمة . ونظرا لأن هذه الطرق والعبارات الخاصة مفيدة جدا كذلك فى توفير معلومات ، عن حالة تنفيذ البرنامج قبل حدوث خطأ منطقى جسيم (انظر مثال ٩ – ٦) فهى مقدمة فى الأقسام من رقم 4 إلى رقم 6 من هذا الفصل .

## ٩ - ٤ الحصول على معلومات تتبع البرنامج

إحدى طرق الحصول على معلومات تتبع البرنامج ، هى وضع سطر تصحيح ( مع وضع D فى عموده السابع ) ، فى بداية كل مقطع من مقاطع جزء الإجراءات ، يحتوى هذا السطر على عبارة DISPLAY . وكلما تم إدخال مقطع .. تنفذ عبارة DISPLAY لتطبع رسالة تخدم كإشارة تتبع للبرنامج .

**مثال ٩ – ٢ :**

```
   SOURCE-COMPUTER. IBM-3081   WITH DEBUGGING MODE.
   . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
   PROCEDURE DIVISION.
D      DISPLAY "PROGRAM EXECUTION BEGINNING"
   . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
   PARA-ONE.
D      DISPLAY "PARA-ONE BEING ENTERED"
   . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
   PARA-TWO.
D      DISPLAY "PARA-TWO BEING ENTERED"
   . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

عندما يتم تصحيح البرنامج .. يمكن حذف WITH DEBUGGING MODE من قطع كمبيوتر المصدر . وتعامل كل الأسطر الموجودة D فى عمودها السابع عند ذلك كتعليقات ، وهذا يحذف مخرجات التتبع .
ويقدم كوبل IBM OS / VS عبارة خاصة ، تهدف كذلك إلى إنتاج تتبع البرنامج .

READY TRACE

والتى تلغى عمل التتبع . يمكن تنفيذ التتبع معد READY TRACE ، وإعادة إعداد التتبع RESET TRACE ، لأى عدد مطلوب من المرات فى البرنامج .

**مثال ٩ - ٢ :**

يستخدم جزء الإجراءات فى مثال ٩ - ٢ عبارة READY TRACE ، ويوضح عينة للمخرجات فى نهايته. وعبارات /READY RESET TRACE فى كل المخرجات .. ففى كوبل IBM OS / VS تطبع READY TRACE مخرجاتها على ملف نظام خاص SYSOUT . وعلى هذا ، يجب أن توجد عبارة لغة تحكم العمل لهذا الملف، عندما تستخدم READY TRACE ( وإلا فسوف ينتهى البرنامج نهاية غير طبيعية ) . لاحظ أن هذا التتبع يطبع اسم مقطع الكوبل ، أول عبارة فى المقطع المنفذ مباشرة. وتحت لغات تحكم عمل مختلفة.. يمكن أن يطبع رقم السطر الذى ينتجه المترجم بدلا من ذلك.

```
        PROCEDURE DIVISION.

        *---------------------------------------
             READY TRACE
        *---------------------------------------

             MOVE "NO " TO END-OF-CARDS-SWITCH
             MOVE ZERO TO TOTAL-SALES

             OPEN    INPUT                 CARD-FILE
                     OUTPUT                PRINT-FILE

             PERFORM 100-INPUT-A-RECORD

             PERFORM 200-PRODUCESALES-LISTING
                 UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"

             WRITE PRINT-LINE
                 FROM SALES-TOTAL-LINE

             CLOSE   CARD-FILE
                     PRINT-FILE
             STOP RUN
             .

        100-INPUT-A-RECORD.

             READ CARD-FILE RECORD
                 INTO WORKING-SALES-CARD
                 AT END
                     MOVE "YES" TO END-OF-CARDS-SWITCH
             .

        200-PRODUCESALES-LISTING.

             ADD WORKING-SALES-AMOUNT TO TOTAL-SALES

             MOVE WORKING-SALES-ID        TO LISTING-SALES-ID
             MOVE WORKING-CUSTOMER-ID     TO LISTING-CUSTOMER-ID
             MOVE WORKING-SALES-AMOUNT    TO LISTING-SALES-AMOUNT

             WRITE PRINT-LINE
                 FROM SALES-LISTING-LINE

             PERFORM 100-INPUT-A-RECORD
```

شكل ( ٩ - ٢ )

```
100-INPUT-A-RECORD ,200-PRODUCESALES-LISTING ,100-INPUT-A-RECORD ,200-PRODUCESALES-LISTING ,
100-INPUT-A-RECORD ,200-PRODUCESALES-LISTING ,100-INPUT-A-RECORD ,200-PRODUCESALES-LISTING ,
100-INPUT-A-RECORD ,200-PRODUCESALES-LISTING ,100-INPUT-A-RECORD ,200-PRODUCESALES-LISTING ,
100-INPUT-A-RECORD ,
```

شكل (٩ - ٢) تكملة

وهناك ملحق قوى لعبارات READY/RESET TRACE ميسر عن طريق التوسع فى كوبل IBM OS/VS عن IMB النمطى ANS التالى :

```
ON {integer-1   } [AND EVERY {integer-2   }] [UNTIL {integer-3   }]
   {identifier-1}            {identifier-2}          {identifier-3}
   imperative statement
[ELSE
      imperative statement ]
```

مثال ٩ - ٤ :

```
 SOURCE-COMPUTER. IBM-3081  WITH DEBUGGING MODE.
 . . . . . . . . . . . . . . . . . . . . . . . . . . . .
 PROCEDURE DIVISION.
     . . . . . . . . . . . . . . . . . . . . . . . . . .
     PERFORM PARA-A
         UNTIL SOME-CONDITION-OR-OTHER
     . . . . . . . . . . . . . . . . . . . . . . . . . .
 PARA-A.
D    ON 1 AND EVERY 2 UNTIL 9
D        READY TRACE
D    ELSE
D        RESET TRACE
D    .
     . . . . . . . . . . . . . . . . . . . .
```

تنفذ READY TRACE فى أول مرة ، وثالث مرة ، وخامس مرة ، وسابع مرة ، وتاسع مرة يدخل فيها المقطع PARA-A . وتنفذ RESET TRACE ( بدلاً من تنفيذ READY TRACE ) لكل مرات التنفيذ الأخرى للمقطع . لاحظ استخدام D فى العمود رقم 7 فى كل عبارات التصحيح (بما فى ذلك النقطة المرتبة فى نهاية عبارة ON ) . وعلى هذا .. تترجم هذه العبارات بواسطة المترجم ، عندما يتحدد WITH DEBUGGING MODE فقط فى مقطع كمبيوتر المصدر .

وفيما يلى أمثلة أكثر لعبارات ON صحيحة :

- ON 3 AND EVERY 5 READY TRACE ELSE RESET TRACE.
- ON 1 READY TRACE ELSE RESET TRACE.
- ON 1 AND EVERY 2 DISPLAY INPUT-RECORD.
- ON 1 AND EVERY 2

```
  ON 1 AND EVERY 2
      DISPLAY INPUT-RECORD
      READY TRACE
  ELSE
      RESET TRACE
```

## ٩ - ٥ اخراج عناصر بيانات أثناء تنفيذ البرنامج

أبسط طريقة للحصول على معلومات نفايا ذاكرة ديناميكية dynamic memory dump هى إدخال أسطر تصحيح بعبارات DISPLAY فى البرنامج . وعندما تنفذ عبارة DISPLAY .. فإنها تطبع محتويات عنصر البيانات المحدد . ويعد عرض DISPLAY كل سجل مدخلات بعد عبارة READ ، وكل سجل مخرجات قبل عبارة WRITE هو فكرة طيبة .

مثال ٩ - ٥ :

```
 SOURCE-COMPUTER. IBM-3081   WITH DEBUGGING MODE.
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
 PROCEDURE DIVISION.
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
     READ INPUT-FILE
         AT END MOVE "YES" TO END-FILE-SW
 .
D    DISPLAY "INPUT RECORD IS:" INPUT-REC
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
D    DISPLAY "OUTPUT RECORD IS:" OUTPUT-REC
     WRITE OUTPUT-REC
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
     ADD 1 TO SAMPLE-COUNTER
D    DISPLAY "SAMPLE-COUNTER AFTER INCREMENTING:" SAMPLE-COUNTER
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

لاحظ استخدام الثوابت غير العددية فى عبارات DISPLAY فى تسمية مخرجات التصحيح . وتوفر هذه الأسماء معلومات تتبع برنامج كذلك ؛ حيث إنها تحدد موقع تنفيذ الكمبيوتر لتعليمات البرنامج ، عندما تنتج مخرجات DISPLAY . ويمكن لمخرجات العبارات السابقة أن تشبه ما يلى :

```
INPUT RECORD IS:0003A789108264
OUTPUT RECORD IS:172A12 CARTONSB97634900000
SAMPLE-COUNTER AFTER INCREMENTING:0007
```

تذكر : عند استخدام DISPLAY فى كوبل IBM OS/VS : يجب تقديم عبارة لغة تحكم عمل لتعريف ملف الطابع SYSOUT ، وإلا فسوف ينتهى البرنامج نهاية غير طبيعية .

يشمل عديد من صيغ الكوبل عبارة لا تطبع محتويات عنصر البيانات فقط ، بل تطبع كذلك اسم الكوبل لعنصر البيانات ، والتكوين كما يلى :

$$\underline{\text{EXHIBIT}}\begin{Bmatrix}\underline{\text{NAMED}}\\ \underline{\text{CHANGED NAMED}}\\ \underline{\text{CHANGED}}\end{Bmatrix}\begin{Bmatrix}\text{identifier-1}\\ \text{literal-1}\end{Bmatrix}\begin{bmatrix}\text{identifier-2}\\ \text{literal-2}\end{bmatrix}\ldots$$

يطبع EXHIBIT NAMED محتويات كل عنصر بيانات مع اسم الكوبل للعنصر ، فى الصورة :-cobol-name = con tents .

يطبع EXHIBIT CHANGED NAMED اسم الكوبل للعنصر تتبعه محتوياته الحالية ، بشرط أن المحتويات منذ آخر مرة نفذت فيها عبارة EXHIBIT الخاصة هذه .

يطبع EXHIBIT CHANGED محتويات عنصر البيانات المحدد ، بشرط أن تتغير منذ آخر مرة ، نفذت فيها عبارة EXHIBIT هذه ، إذا لم يطبع اسم الكوبل .

وفى كوبل IBM OS/VS .. يجب تقديم عبارة لغة تحكم العمل ؛ لتعريف ملف طباعة SYSOUT عند استخدام أى صورة من صور EXHIBIT .

**مثال ٩ - ٦ :**

أضيفت عبارات EXHIBIT NAMED, READY TRACE الى برنامج شكل ٩ - ٣ ؛ للمساعدة فى تصحيح إنهاء البرنامج نهاية غير طبيعية . توضح مخرجات التصحيح ، شكل (٩ - ٤) كيف خلطت مخرجات EXHIBIT, READY TRACE مع بعضها البعض فى ملف طباعة SYSOUT ، وكيف تبدأ EXHIBIT سطـــرا جديدا تلقائيـا ، إذا لم يكف السطر الواحد كل المعلومات ، وكيف قدمت مخرجات READY TRACE تاريخا مفيدا لما حدث أثناء تنفيذ البرنامج ، قارن التقرير المطبوع فى نهاية شكل ٩ - ٣ مع مخرجات التصحيح ؛ للتأكد من أنك تتبع منطق البرنامج مع تشغيله لكل سجل مدخلات ، تم تنفيذ آخر مقطع (طبقا للتتبع ) هو 200 - PRODUCE - TIME - LISTING ، لاحظ أن الكمبيوتر ذهب على الأقل - إلى نفس بعد عبارة EXHIBIT فى هذا المقطع ، حيث تظهر محتويات WORKING - OVERTIME - HOURS, WORKING - REGULAR- HOURS فى مخرجات التصحيح . نستطيع أن نستخلص أن النهاية غير الطبيعية حدثت أثناء تنفيذ عبارة COMPUTE فى السطر 116 ؛ لان المخرجات من عبارة EXHIBIT توضح أن WORKING - OVER- TIME - HOURS يحتوى على فراغات . ولم تكن هذه النهاية غير الطبيعية كذلك بسبب البرنامج ، بل كانت بسبب بيانات مدخلات غير صحيحة ويجب إعادة تصميم البرنامج بحيث تنقح (يتأكد من صحة) بيانات المدخلات قبل محاولة تشغيلها .

```
       IDENTIFICATION DIVISION.

       PROGRAM-ID. TIMELIST.

       AUTHOR.  LARRY NEWCOMER.
       INSTALLATION.  PENN STATE UNIVERSITY, YORK CAMPUS.

       DATE-COMPILED. MAY  9,1983.

       ENVIRONMENT DIVISION.

       CONFIGURATION SECTION.

       SOURCE-COMPUTER. IBM-3081 WITH DEBUGGING MODE.
       OBJECT-COMPUTER. IBM-3081.

       INPUT-OUTPUT SECTION.

       FILE-CONTROL.

           SELECT CARD-FILE            ASSIGN TO CARDS.
           SELECT PRINT-FILE           ASSIGN TO PRINTER.

       DATA DIVISION.

       FILE SECTION.

       FD  CARD-FILE
           RECORD CONTAINS 80 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .

       01  TIME-CARD-INPUT             PIC X(80).

       FD  PRINT-FILE
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .

       01  PRINT-LINE                  PIC X(132).

       WORKING-STORAGE SECTION.

       01  END-OF-CARDS-SWITCH         PIC X(3).

       01  WORKING-TIME-CARD.

           05  WORKING-NAME            PIC X(20).
           05  WORKING-REGULAR-HOURS   PIC 99.
           05  WORKING-OVERTIME-HOURS  PIC 99.
           05  FILLER                  PIC X(56).

       01  TIME-LISTING-LINE.

           05  LISTING-NAME            PIC X(20).
           05  FILLER                  PIC X(5)       VALUE SPACES.
           05  LISTING-REGULAR-HOURS   PIC 99.
           05  FILLER                  PIC X(5)       VALUE SPACES.
           05  LISTING-OVERTIME-HOURS  PIC 99.
           05  FILLER                  PIC X(5)       VALUE SPACES.
```

شكل ( ٩ - ٣ )

```
           05  LISTING-TOTAL-HOURS        PIC 999.
           05  FILLER                     PIC X(85)      VALUE SPACES.

       01  EMPLOYEE-TOTAL-LINE.

           05  FILLER                     PIC X(25)
                                          VALUE "NUMBER OF EMPLOYEES IS".
           05  NUMBER-OF-EMPLOYEES        PIC 999.
           05  FILLER                     PIC X(104)     VALUE SPACES.

       PROCEDURE DIVISION.

      D    READY TRACE

           MOVE "NO " TO END-OF-CARDS-SWITCH
           MOVE ZERO  TO NUMBER-OF-EMPLOYEES

           OPEN    INPUT                  CARD-FILE
                   OUTPUT                 PRINT-FILE

           PERFORM 100-INPUT-A-RECORD

           PERFORM 200-PRODUCE-TIME-LISTING
               UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"

           WRITE PRINT-LINE
               FROM EMPLOYEE-TOTAL-LINE

           CLOSE   CARD-FILE
                   PRINT-FILE
           STOP RUN
           .

       100-INPUT-A-RECORD.

           READ CARD-FILE RECORD
               INTO WORKING-TIME-CARD
               AT END
                   MOVE "YES" TO END-OF-CARDS-SWITCH
           .
      D    EXHIBIT NAMED WORKING-TIME-CARD
      D                  END-OF-CARDS-SWITCH
      D    .

       200-PRODUCE-TIME-LISTING.

           ADD 1 TO NUMBER-OF-EMPLOYEES

           MOVE WORKING-NAME              TO LISTING-NAME

           MOVE WORKING-REGULAR-HOURS   TO LISTING-REGULAR-HOURS
           MOVE WORKING-OVERTIME-HOURS  TO LISTING-OVERTIME-HOURS

      D    EXHIBIT NAMED WORKING-REGULAR-HOURS
      D                  WORKING-OVERTIME-HOURS
           COMPUTE LISTING-TOTAL-HOURS =  WORKING-REGULAR-HOURS
                                        + WORKING-OVERTIME-HOURS

           WRITE PRINT-LINE
               FROM TIME-LISTING-LINE
```

شكل ( ٩ - ٣ ) تكملة

```
00121
00122              PERFORM 100-INPUT-A-RECORD
00123              .

-------------------------------------------------
PRINTED REPORT:

ABEL FRED              40     03     043
BAKER SUE              30     00     030
CHARLIE CHUCK          40     12     052
PERELMAN BARNEY        40     15     055
PERELMAN MIKE          03     00     003
```

شكل (٩ - ٣) تكملة

```
100-INPUT-A-RECORD ,
WORKING-TIME-CARD = ABEL FRED          4003          END-OF-CARDS-SWITCH
 = NO
200-PRODUCE-TIME-LISTING ,
WORKING-REGULAR-HOURS = 40 WORKING-OVERTIME-HOURS = 03
100-INPUT-A-RECORD ,
WORKING-TIME-CARD = BAKER SUE          3000          END-OF-CARDS-SWITCH
 = NO
200-PRODUCE-TIME-LISTING ,
WORKING-REGULAR-HOURS = 30 WORKING-OVERTIME-HOURS = 00
100-INPUT-A-RECORD ,
WORKING-TIME-CARD = CHARLIE CHUCK      4012          END-OF-CARDS-SWITCH
 = NO
200-PRODUCE-TIME-LISTING ,
WORKING-REGULAR-HOURS = 40 WORKING-OVERTIME-HOURS = 12
100-INPUT-A-RECORD ,
WORKING-TIME-CARD = PERELMAN BARNEY    4015          END-OF-CARDS-SWITCH
 = NO
200-PRODUCE-TIME-LISTING ,
WORKING-REGULAR-HOURS = 40 WORKING-OVERTIME-HOURS = 15
100-INPUT-A-RECORD ,
WORKING-TIME-CARD = PERELMAN MIKE      0300          END-OF-CARDS-SWITCH
 = NO
200-PRODUCE-TIME-LISTING ,
WORKING-REGULAR-HOURS = 03 WORKING-OVERTIME-HOURS = 00
100-INPUT-A-RECORD ,
WORKING-TIME-CARD = TROUBLE TOM        40            END-OF-CARDS-SWITCH
 = NO
200-PRODUCE-TIME-LISTING ,
WORKING-REGULAR-HOURS = 40 WORKING-OVERTIME-HOURS =
```

شكل ( ٩ - ٤ )

## ٩ - ٦ التوضيحات وعبارة يستخدم للتصحيح

يقدم الكوبل النمطى ANS توضيحات DECLARATIVES لأجزاء تصحيح خاصة ، ويجب أن تكتب هذه التوضيحات فى بداية جزء الإجراءات . ومثل هذه الاجزاء فريدة ، فى أنها لا تنفذ كجزء من منطق البرنامج المعتاد ، وإنما تنفذ إذا ما أثيرت

بواسطه أحداث خاصة محددة فقط .

مثال ٩ - ٧ :

```
PROCEDURE DIVISION.
DECLARATIVES.
DECLARATIVE-SECTION-1 SECTION.
. . . . . . . . . . . . . . . . . . . . . . . . .
DECLARATIVE-SECTION-2 SECTION.
. . . . . . . . . . . . . . . . . . . . . . . . . .
END DECLARATIVES.
. . . (normal PROCEDURE DIVISION statements)
```

... (عبارة عادية من عبارات جزء الإجراءات)

لاحظ العنوان DECLARATIVES ، وأخر سطر END DECLARATIVES ( المكتوبة فى المنطقة A ) .

يجب أن يبدأ كل قسم توضيحات DECLARATIVE SECTION بعبارة USE ، والتى تحدد متى ينفذ قسم التوضيحات . يعرض الشكل العام لتوضيحات التصحيح DEBUGGING DECLARATIVES فى شكل ٩ - ٥ .

```
USE FOR DEBUGGING
       { [ALL REFERENCES OF] identifier-1 }  [ [ALL REFERENCES OF] identifier-2 ]
    ON { file-name-1                      }  [ file-name-2                      ] . . .
       { procedure-name-1                 }  [ procedure-name-2                 ]
       { ALL PROCEDURES                   }  [ ALL PROCEDURES                   ]
```

شكل ( ٩ - ٥ )

مثال ٩ - ٨ :

```
PROCEDURE DIVISION.
DECLARATIVES.
CHECK-COUNTER SECTION.
    USE FOR DEBUGGING
       ON ALL REFERENCES OF NUMBER-CUSTOMERS-BILLED
    .
    EXHIBIT NAMED NUMBER-CUSTOMERS-BILLED
    EXHIBIT NAMED CURRENT-CUSTOMER-ID
    .
END DECLARATIVES.
 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

استخدم هنا جزء ON ALL REFERENCE OF : identifier-1 . وينفذ القسم (فى هذه الحالة .. عبارتى EXHIBIT NAMED ) قبل كل كتابة WRITE معرف identifier-1 مباشرة ، وبعد أى عبارة كوبل تحتوى – صراحة ومباشرة – على identifier-1 .

وفيما يلى بدائل أخرى لـ USE FOR DEBUGGING .

- ON identifier-1 (ON SAMPLE-OUTPUT-RECORD)

عادة ما يكون identifier-1 اسم سجل كوبل للملف . مخرجات تنفذ التوضيحات DECLARATIVE قبل كل عبارة WRITE فى الملف مباشرة (مثل قبل عبارة WRITE SAMPLE - OUTPUT - RECORD ) .

- ON file-name-1 (ON EMPLOYEE-MASTER-FILE)

تنفذ التوضيحات DECLARAITVE بعد أى عبارة OPEN أو عبارة CLOSE ، أو عبارة READ للملف المحدد .

- ON procedure-name-1 (ON 200-PRODUCE-MAILING-LABELS)

procedure - name - 1 هو اسم مقطع أو اسم قسم فى جزء الإجراءات ، ينفذ التوضيحات قبل كل تنفيذ للإجراء المسمى .

- ON ALL PROCEDURES

تنفذ التوضيحات قبل كل تنفيذ لكل المقاطع وكل الأقسام ( باستثناء أقسام التوضيحات نفسها ) .

وكما هو الحال مع أسطر التصحيح الأخرى .. تترجم توضيحات التصحيح DEBUGGING DECLARATIVES إلى برنامج هدف بلغة الآلة إذا حدد WITH DEBUGGING MODE فى مقطع برنامج المصدر فقط .

عندما تنفذ توضيحات التصحيح .. يضع النظام معلومات قيمة فى منطقة بيانات خاصة ، تسمى DEBUG - ITEM . وكلمة DEBUG - ITEM هى كلمة كوبل محجوزة ، لا يعرفها المبرمج فى جزء البيانات ، وهى لها تخطيط الكوبل المحدد فى شكل ٩ – ٦ .

```
01  DEBUG-ITEM.
    05  DEBUG-LINE        PIC X(6).
    05  FILLER            PIC X       VALUE SPACES.
    05  DEBUG-NAME        PIC X(30).
    05  FILLER            PIC X       VALUE SPACES.
    05  DEBUG-SUB-1       PIC S9999   SIGN IS LEADING SEPARATE CHARACTER.
    05  FILLER            PIC X       VALUE SPACES.
    05  DEBUG-SUB-2       PIC S9999   SIGN IS LEADING SEPARATE CHARACTER.
    05  FILLER            PIC X       VALUE SPACES.
    05  DEBUG-SUB-3       PIC S9999   SIGN IS LEADING SEPARATE CHARACTER.
    05  FILLER            PIC X       VALUE SPACES.
    05  DEBUG-CONTENTS    PIC X(n).
```

شكل ( ٩ – ٦ )

تكون محتويات حقول المستوى 05 كما يلى :

١ - يوضع DEBUG - LINE لرقم سطر ينتجه المترجم للعبارة التى تسببت فى تنفيذ DFCLARATIVE .

٢ - يوضع DEBUG - NAME لأول 30 رمز من الاسم الذى تسبب فى تنفيذ DECLARATIVES .

٣ - توضع DEBUG - SUB - 1 ,DEBUG - SUB - 2 ,DEBUG - SUB - 3 لقيم الدليل الحاليه للعنصر DEBUG-NAME، إذا كان جزءا من جدول (انظر الفصل العاشر) ، وإلا.. فسوف تترك هذه الأجزاء من DEBUG - ITEM فارغة.

٤ - توضع DEBUG - CONTENTS لمحتويات عنصر البيانات (الذى يكون اسمه فى DEBUG - NAME ) الذى تسبب فى تنفيذ DECLARATIVES فإذا كان DECLARATIVE قد نفذ بسبب USE FOR DEBUGGING ON Procc-dure. name .. فـإن DEBUG - NAME تحتوى على اسم الإجراءات ، ويوضع DEBUG - CONTENTS لعبارة إنجليزية تصف نوع الإجراء المعنى .

يحتوى DEBUG - ITEM على معلومات مفيدة جدا بحيث يكون استخدام DISPLAY DEBUG - ITEM أو EXHIB-IT DEBUG - ITEM بسيطة هو كل المطلوب فى قسم التوضيحات .

**مثال ٩ - ٩ :**

نعيد اعتبار النهاية غير الطبيعية فى مثال ٩ - ٦ ، مع استخدام التوضيحات كوسيلة تصحيح فى هذه المرة . جزء الإجراءات الجديد موضح فى شكل ٩ - ٧ ، مع تحديد الأسطر من ٧٤ الى ٨٨ للتوضيحات . عندما يحدد عديد من الشروط فى جزء ON واحد ، يتسبب حدوث أى منها فى تنفيذ التوضيحات . لاحظ أن كل التوضيحات تكون EXHIBIT DEBUG - ITEM . تبين عينة المخرجات ، شكل ٩ - ٨ ، كم المعلومات الذى تظهره هذه العبارة البسيطة . ( هيئة شكل ٩ - ٨ تتبع شكل ٩ - ٦ ، مع فراغات فى DEBUG - SUB ومع PERFORM LOOP بأنها DEBUG - CONTENTS لـ 200 - PRODUCE - TIME - LISTING ) .. نستدل أن البرنامج ذهب على الأقل حتى السطر ١٢٤ ( يظهر أخر سطر فى مخرجات التصحيح ، ويحتوى WORKING - OVERTIME - HOURS عند هذه النقطة ، على فراغات . وحيث إن السطر ١٢٥ فى برنامج الكوبل فارغ ويستخدم السطر ١٢٦ WORKING - OVERTIME - HOURS ( المحتوى على فراغات ) فى حسابات .. تتضح مشكلة الإنهاء غير الطبيعى ( انظر مثال ٩ - ٦ ) .

```
00072          PROCEDURE DIVISION.

00074          DECLARATIVES.
00075
00076          IDENTIFY-ABEND SECTION.
00077
00078              USE FOR DEBUGGING
00079                  ON ALL REFERENCES OF WORKING-TIME-CARD
00080                     ALL REFERENCES OF WORKING-REGULAR-HOURS
00081                     ALL REFERENCES OF WORKING-OVERTIME-HOURS
00082                     ALL REFERENCES OF 200-PRODUCE-TIME-LISTING
0Q083
00084
```

**شكل (٩ - ٧)**

```
           EXHIBIT NAMED DEBUG-ITEM
           .

       END DECLARATIVES.


           MOVE "NO " TO END-OF-CARDS-SWITCH
           MOVE ZERO  TO NUMBER-OF-EMPLOYEES

           OPEN    INPUT                CARD-FILE
                   OUTPUT               PRINT-FILE

           PERFORM 100-INPUT-A-RECORD

           PERFORM 200-PRODUCE-TIME-LISTING
               UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"

           WRITE PRINT-LINE
               FROM EMPLOYEE-TOTAL-LINE

           CLOSE   CARD-FILE
                   PRINT-FILE
           STOP RUN
           .

       100-INPUT-A-RECORD.

           READ CARD-FILE RECORD
               INTO WORKING-TIME-CARD
               AT END
                   MOVE "YES" TO END-OF-CARDS-SWITCH


       200-PRODUCE-TIME-LISTING.

           ADD 1 TO NUMBER-OF-EMPLOYEES

           MOVE WORKING-NAME            TO LISTING-NAME
           MOVE WORKING-REGULAR-HOURS   TO LISTING-REGULAR-HOURS
           MOVE WORKING-OVERTIME-HOURS  TO LISTING-OVERTIME-HOURS

           COMPUTE LISTING-TOTAL-HOURS = WORKING-REGULAR-HOURS
                                       + WORKING-OVERTIME-HOURS


           WRITE PRINT-LINE
               FROM TIME-LISTING-LINE

           PERFORM 100-INPUT-A-RECORD
```

شكل (٩ - ٧) تكملة

```
DEBUG-ITEM = 000112 WORKING-TIME-CARD            ABEL FRED              4003

DEBUG-ITEM = 000099 200-PRODUCE-TIME-LISTING     PERFORM LOOP
DEBUG-ITEM = 000123 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000124 WORKING-OVERTIME-HOURS       03
DEBUG-ITEM = 000126 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000126 WORKING-OVERTIME-HOURS       03
DEBUG-ITEM = 000112 WORKING-TIME-CARD            BAKER SUE              3000

DEBUG-ITEM = 000099 200-PRODUCE-TIME-LISTING     PERFORM LOOP
DEBUG-ITEM = 000123 WORKING-REGULAR-HOURS        30
DEBUG-ITEM = 000124 WORKING-OVERTIME-HOURS       00
DEBUG-ITEM = 000126 WORKING-REGULAR-HOURS        30
DEBUG-ITEM = 000126 WORKING-OVERTIME-HOURS       00
DEBUG-ITEM = 000112 WORKING-TIME-CARD            CHARLIE CHUCK          4012

DEBUG-ITEM = 000099 200-PRODUCE-TIME-LISTING     PERFORM LOOP
DEBUG-ITEM = 000123 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000124 WORKING-OVERTIME-HOURS       12
DEBUG-ITEM = 000126 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000126 WORKING-OVERTIME-HOURS       12
DEBUG-ITEM = 000112 WORKING-TIME-CARD            PERELMAN BARNEY        4015

DEBUG-ITEM = 000099 200-PRODUCE-TIME-LISTING     PERFORM LOOP
DEBUG-ITEM = 000123 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000124 WORKING-OVERTIME-HOURS       15
DEBUG-ITEM = 000126 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000126 WORKING-OVERTIME-HOURS       15
DEBUG-ITEM = 000112 WORKING-TIME-CARD            PERELMAN MIKE          0300

DEBUG-ITEM = 000099 200-PRODUCE-TIME-LISTING     PERFORM LOOP
DEBUG-ITEM = 000123 WORKING-REGULAR-HOURS        03
DEBUG-ITEM = 000124 WORKING-OVERTIME-HOURS       00
DEBUG-ITEM = 000126 WORKING-REGULAR-HOURS        03
DEBUG-ITEM = 000126 WORKING-OVERTIME-HOURS       00
DEBUG-ITEM = 000112 WORKING-TIME-CARD            TROUBLE TOM            40

DEBUG-ITEM = 000099 200-PRODUCE-TIME-LISTING     PERFORM LOOP
DEBUG-ITEM = 000123 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000124 WORKING-OVERTIME-HOURS
```

شكل ( ٩ - ٨ )

## اسئلة مراجعة

٩ - ١ ماهو اختبار البرنامج ؟

٩ - ٢ ناقش ثلاثة أنواع من الأخطاء التى يمكن أن تحدث أثناء اختبار البرنامج .

٩ - ٣ ماهى رسائل الخطأ التشخيصية ؟

٩ - ٤ ناقش الاختلافات بين الأخطاء التكوينية من المستوى W والمستوى E .

٩ - ٥ ماهى نفايا الذاكرة ؟ ميز بين النفايا السادسة عشرية والنفايا الرمزية .

٩ - ٦ ماهو تتبع البرنامج ؟ كيف يكون مفيدا فى التصحيح ؟

٩ – ٧ ما هى المعلومات التى تقدم بصورةه تقليدية فى النفايا الرمزية ؟

٩ – ٨ كيف يمكن استخدام عبارات DISPLAY فى التصحيح ؟

٩ – ٩ صف آثار RESET TRACE, READY TRACE .

٩ – ١٠ صف تكوين واستخدام عبارة ON .

٩ – ١١ ناقش تكوين واستخدام عبارة EXHIBIT .

٩ – ١٢ ما شكل المخرجات من EXHIBIT NAMED

٩ – ١٣ ما هى التوضيحات DECLARATIVES ؟

٩ – ١٤ كيف تختلف التوضيحات عن العبارات الأخرى فى جزء الإجراءات ؟

٩ – ١٥ ناقش تأثير مقطع كمبيوتر المصدر على التوضيحات ، والأسطر التى بها D فى عمودها السابع

٩ – ١٦ ناقش تأثيرات ما يلى فى عبارة USE FOR DEBUGGING

(أ) ON ALL REFERENCES

(ب) ON file - name

(جـ) ON procedure - name

(د) ON ALL PROCEDURES

٩ – ١٧ ناقش محتويات DEBUG - ITEM ومدى فائدتها .

## مسائل محلولة

٩ – ١٨ صحح البرنامج التالى ، الذى يحتوى على أخطاء منطقية غير جسيمة . معطى لك قائمة ببرنامج المصدر (شكل ٩ – ٩)، ومخرجات اختبار (شكل ٩ – ١٠) ، ومخرجات تصحيح من DEBUGGING DECLARATIVES (شكل ٩ – ١١) . المخرجات غير صحيحة لما يلى :

(١) عدد العاملين يجب أن يكون 006 وليس 012 .

(٢) لا يوجد عامل يتساوى حاصل جمع عدد ساعات عمله المعتادة على عدد ساعات عمله وقتا اضافيا ، مع إجمالى ساعات عمله (أى أن أعمدة شكل ٩ – ١٠ لا تجمع بطريقة صحيحة) .

```
        IDENTIFICATION DIVISION.

        PROGRAM-ID. TIMELIST.

        AUTHOR.  LARRY NEWCOMER.
        INSTALLATION.  PENN STATE UNIVERSITY--YORK CAMPUS.

        DATE-WRITTEN.  MAY 1983.
        DATE-COMPILED. MAY  9,1983.
       *    TIMELIST PRINTS ONE LINE FOR EACH EMPLOYEE SHOWING:
       *    NUMBER OF REGULAR HOURS WORKED, NUMBER OF OVERTIME
       *    HOURS WORKED, AND TOTAL HOURS WORKED. AT THE END OF THE
       *    REPORT, IT ALSO PRINTS A COUNT OF THE NUMBER OF
       *    EMPLOYEES PROCESSED.  INPUT IS FROM A DECK OF TIME
       *    CARDS, KEPT IN SEQUENCE BY EMPLOYEE NAME.

        ENVIRONMENT DIVISION.
```

شكل (٩ – ٩)

```
        CONFIGURATION SECTION.

        SOURCE-COMPUTER. IBM-3081 WITH DEBUGGING MODE.
        OBJECT-COMPUTER. IBM-3081.
    INPUT-OUTPUT SECTION.

    FILE-CONTROL.

        SELECT CARD-FILE               ASSIGN TO CARDS.
        SELECT PRINT-FILE              ASSIGN TO PRINTER.

    DATA DIVISION.

    FILE SECTION.

    FD  CARD-FILE
        RECORD CONTAINS 80 CHARACTERS
        LABEL RECORDS ARE OMITTED
        .

    01  TIME-CARD-INPUT                PIC X(80).

    FD  PRINT-FILE
        RECORD CONTAINS 132 CHARACTERS
        LABEL RECORDS ARE OMITTED
        .

    01  PRINT-LINE                     PIC X(132).

    WORKING-STORAGE SECTION.

    01  END-OF-CARDS-SWITCH            PIC X(3).

    01  WORKING-TIME-CARD.

        05  WORKING-NAME               PIC X(20).
        05  WORKING-REGULAR-HOURS      PIC 99.
        05  WORKING-OVERTIME-HOURS     PIC 99.
        05  FILLER                     PIC X(56).

    01  TIME-LISTING-LINE.

        05  LISTING-NAME               PIC X(20).
        05  FILLER                     PIC X(5)      VALUE SPACES.
        05  LISTING-REGULAR-HOURS      PIC 99.
        05  FILLER                     PIC X(5)      VALUE SPACES.
        05  LISTING-OVERTIME-HOURS     PIC 99.
        05  FILLER                     PIC X(5)      VALUE SPACES.
        05  LISTING-TOTAL-HOURS        PIC 999.
        05  FILLER                     PIC X(85)     VALUE SPACES.

    01  EMPLOYEE-TOTAL-LINE.

        05  FILLER                     PIC X(25)
                                       VALUE "NUMBER OF EMPLOYEES IS".
        05  NUMBER-OF-EMPLOYEES        PIC 999.
        05  FILLER                     PIC X(104)    VALUE SPACES.
    PROCEDURE DIVISION.
```

شكل ( ٩ - ٩ ) تكملة

```
    DECLARATIVES.

    CHECK-LOGIC SECTION.

        USE FOR DEBUGGING
            ON ALL REFERENCES OF NUMBER-OF-EMPLOYEES
               ALL REFERENCES OF WORKING-REGULAR-HOURS
               ALL REFERENCES OF WORKING-OVERTIME-HOURS
               ALL REFERENCES OF LISTING-TOTAL-HOURS

        EXHIBIT NAMED DEBUG-ITEM


  END DECLARATIVES.

      MOVE "NO " TO END-OF-CARDS-SWITCH
      MOVE ZERO TO NUMBER-OF-EMPLOYEES

      OPEN    INPUT                 CARD-FILE
              OUTPUT                PRINT-FILE

      PERFORM INPUT-A-RECORD

      PERFORM PRODUCE-TIME-LISTING
          UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"

      WRITE PRINT-LINE
          FROM EMPLOYEE-TOTAL-LINE

      CLOSE   CARD-FILE
              PRINT-FILE
      STOP RUN

  INPUT-A-RECORD.

      READ CARD-FILE
          INTO WORKING-TIME-CARD
          AT END
              MOVE "YES" TO END-OF-CARDS-SWITCH
              .

  PRODUCE-TIME-LISTING.

*             INCREMENT COUNTER EACH TIME AN EMPLOYEE IS PROCESSED

      ADD 2 TO NUMBER-OF-EMPLOYEES

*             MOVE DATA FROM INPUT CARD TO AREA WHERE LINE IS BUILT

      MOVE WORKING-NAME           TO LISTING-NAME
      MOVE WORKING-REGULAR-HOURS  TO LISTING-REGULAR-HOURS
      MOVE WORKING-OVERTIME-HOURS TO LISTING-OVERTIME-HOURS

*             CALCULATE TOTAL HOURS BY ADDING REGULAR, OVERTIME
*             NOTE THAT RESULT IS PLACED IN TIME-LISTING-LINE.

      ADD WORKING-REGULAR-HOURS WORKING-REGULAR-HOURS
          GIVING LISTING-TOTAL-HOURS

      WRITE PRINT-LINE
          FROM TIME-LISTING-LINE

      PERFORM INPUT-A-RECORD
```

شكل (٩ - ٩) تكملة

```
ABEL FRED                 40      03      080
BAKER SUE                 30      00      060
CHARLIE CHUCK             40      12      080
PERELMAN BARNEY           40      15      080
PERELMAN MIKE             03      00      006
TROUBLE TOM               40      10      080
NUMBER OF EMPLOYEES IS    012
```

شكل ( ٩ - ١٠ )

```
DEBUG-ITEM = 000097 NUMBER-OF-EMPLOYEES          000
DEBUG-ITEM = 000127 NUMBER-OF-EMPLOYEES          002
DEBUG-ITEM = 000132 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000133 WORKING-OVERTIME-HOURS       03
DEBUG-ITEM = 000138 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000138 LISTING-TOTAL-HOURS          080
DEBUG-ITEM = 000127 NUMBER-OF-EMPLOYEES          004
DEBUG-ITEM = 000132 WORKING-REGULAR-HOURS        30
DEBUG-ITEM = 000133 WORKING-OVERTIME-HOURS       00
DEBUG-ITEM = 000138 WORKING-REGULAR-HOURS        30
DEBUG-ITEM = 000138 LISTING-TOTAL-HOURS          060
DEBUG-ITEM = 000127 NUMBER-OF-EMPLOYEES          006
DEBUG-ITEM = 000132 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000133 WORKING-OVERTIME-HOURS       12
DEBUG-ITEM = 000138 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000138 LISTING-TOTAL-HOURS          080
DEBUG-ITEM = 000127 NUMBER-OF-EMPLOYEES          008
DEBUG-ITEM = 000132 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000133 WORKING-OVERTIME-HOURS       15
DEBUG-ITEM = 000138 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000138 LISTING-TOTAL-HOURS          080
DEBUG-ITEM = 000127 NUMBER-OF-EMPLOYEES          010
DEBUG-ITEM = 000132 WORKING-REGULAR-HOURS        03
DEBUG-ITEM = 000133 WORKING-OVERTIME-HOURS       00
DEBUG-ITEM = 000138 WORKING-REGULAR-HOURS        03
DEBUG-ITEM = 000138 LISTING-TOTAL-HOURS          006
DEBUG-ITEM = 000127 NUMBER-OF-EMPLOYEES          012
DEBUG-ITEM = 000132 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000133 WORKING-OVERTIME-HOURS       10
DEBUG-ITEM = 000138 WORKING-REGULAR-HOURS        40
DEBUG-ITEM = 000138 LISTING-TOTAL-HOURS          080
```

شكل ( ٩ - ١١ )

فيما يلى إحدى طرق تصحيح البرنامج :

١ – نبدأ بعداد عدد العاملين . مخرجات التصحيح EXHIBIT DEBUG - ITEM على ALL REFERENCES OF NUM- BER - OF EMPLOYEES .. نجد أن NUMBER - OF - EMPLOYEES قد وضع صفرا فى السطر ٩٧ .

٢ – الإشارة التالية لـ NUMBER - OF EMPLOYEES فى السطر ١٢٧ ، عندما أصبح 002 . كيف حدث قفز من 000 إلى 002 دون أخذ القيمة 001 .

٣ – بالنظر الى السطر ١٢٧ فإننا نعرف .

```
ADD 2 TO NUMBER-OF-EMPLOYEES
```

يفترض أن يزيد هذا السطر من العداد فى كل مرة يتم إدخال سجل عامل جديد . من الواضح أننا يجب أن نجمع «1» ، وليس «2». وما حدث هنا هو خطأ فى الكتابة فى برنامج الكوبل . لاحظ ، على أيه حال .. لم يعترض هذا الخطأ المترجم (فتكوين السطر ١٢٧ صحيح ) ، ولم يتسبب فى إنهاء البرنامج نهاية غير طبيعية . يجب أن يجد المبرمج مثل هذا الخطأ أثناء اختباره البرنامج .

٤ – نعود الآن إلى عمود إجمالى ساعات العمل . توضح مخرجات التصحيح محتويات WORKING - REGULAR - HOURS ، ومحتويات WORKING - OVERTIME - HOURS أثناء تنفيذ عبارة MOVE فى السطرين ١٣٢ ، ١٣٣. مثال ذلك توجد ساعات العمل المعتادة لأول عامل ، وهى 40 وساعات العمل لوقت إضافى هى03 .

٥ – الإشاره التالية إلى WORKING - REGULAR - HOURS فى السطر ١٣٨ ، والتى تشير كذلك إلى LISTING - TOTAL - HOURS ، الحقل غير الصحيح . لاحظ أن WORKING - OVERTIME - HOURS لا يشار إليه فى السطر ١٣٨، وأن LISTING - TOTAL - HOURS تكون دائما ضعف القيمة الموجودة فى WORKING - REG- ULAR - HOURS .

٦ – بفحص العبارة الموجودة فى السطر ١٣٨ نجد أن :

```
ADD WORKING-REGULAR-HOURS   WORKING-REGULAR-HOURS
    GIVING LISTING-TOTAL-HOURS
```

بدلا من العبارة المقصودة ، وهى :

```
ADD WORKING-REGULAR-HOURS   WORKING-OVERTIME-HOURS
    GIVING LISTING-TOTAL-HOURS
```

وهذا خطأ آخر لا يتسبب فى حدوث خطأ تكوينى ، ولاينهى البرنامج نهاية غير طبيعية .

٩ – ١٩ صحح البرنامج التالى ، الذى يحتوى على دورة لا نهائية . قائمة برنامج المصدر ، ومخرجات الاختبار ، ومخرجات التصحيح مبينة فى الأشكال : (٩ – ١٢) ، (٩ – ١٣) ، (٩ – ١٤) على التوالى .

```
       IDENTIFICATION DIVISION.

       PROGRAM-ID. TIMELIST.

       AUTHOR.  LARRY NEWCOMER.
       INSTALLATION.  PENN STATE UNIVERSITY--YORK CAMPUS.

       DATE-WRITTEN.  MAY 1983.
       DATE-COMPILED. MAY  9,1983.
      *    TIMELIST PRINTS ONE LINE FOR EACH EMPLOYEE SHOWING:
      *    NUMBER OF REGULAR HOURS WORKED, NUMBER OF OVERTIME
      *    HOURS WORKED, AND TOTAL HOURS WORKED. AT THE END OF THE
      *    REPORT, IT ALSO PRINTS A COUNT OF THE NUMBER OF
      *    EMPLOYEES PROCESSED.  INPUT IS FROM A DECK OF TIME
      *    CARDS, KEPT IN SEQUENCE BY EMPLOYEE NAME.
       ENVIRONMENT DIVISION.

       CONFIGURATION SECTION.

       SOURCE-COMPUTER. IBM-3081 WITH DEBUGGING MODE.
       OBJECT-COMPUTER. IBM-3081.

       INPUT-OUTPUT SECTION.

       FILE-CONTROL.

           SELECT CARD-FILE             ASSIGN TO CARDS.
           SELECT PRINT-FILE            ASSIGN TO PRINTER.

       DATA DIVISION.

       FILE SECTION.

       FD  CARD-FILE
           RECORD CONTAINS 80 CHARACTERS
           LABEL RECORDS ARE OMITTED


       01  TIME-CARD-INPUT              PIC X(80).

       FD  PRINT-FILE
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED


       01  PRINT-LINE                   PIC X(132).

       WORKING-STORAGE SECTION.

       01  END-OF-CARDS-SWITCH          PIC X(3).

       01  WORKING-TIME-CARD.

           05  WORKING-NAME             PIC X(20).
           05  WORKING-REGULAR-HOURS    PIC 99.
           05  WORKING-OVERTIME-HOURS   PIC 99.
           05  FILLER                   PIC X(56).
```

شكل ( ٩ - ١٢ )

```
          01  TIME-LISTING-LINE.

              05  LISTING-NAME               PIC X(20).
              05  FILLER                     PIC X(5)       VALUE SPACES.
              05  LISTING-REGULAR-HOURS      PIC 99.
              05  FILLER                     PIC X(5)       VALUE SPACES.
              05  LISTING-OVERTIME-HOURS     PIC 99.
              05  FILLER                     PIC X(5)       VALUE SPACES.
              05  LISTING-TOTAL-HOURS        PIC 999.
              05  FILLER                     PIC X(85)      VALUE SPACES.

          01  EMPLOYEE-TOTAL-LINE.

              05  FILLER                     PIC X(25)
                                             VALUE "NUMBER OF EMPLOYEES IS".
              05  NUMBER-OF-EMPLOYEES        PIC 999.
              05  FILLER                     PIC X(104)     VALUE SPACES.

          PROCEDURE DIVISION.

          DECLARATIVES.

          CHECK-LOGIC SECTION.

              USE FOR DEBUGGING
                  ON ALL REFERENCES OF NUMBER-OF-EMPLOYEES
                     ALL REFERENCES OF WORKING-REGULAR-HOURS
                     ALL REFERENCES OF WORKING-OVERTIME-HOURS
                     ALL REFERENCES OF LISTING-TOTAL-HOURS
              .
              EXHIBIT NAMED DEBUG-ITEM
              .

          END DECLARATIVES.

         D    READY TRACE

              MOVE "NO " TO END-OF-CARDS-SWITCH
              MOVE ZERO TO NUMBER-OF-EMPLOYEES

              OPEN    INPUT                  CARD-FILE
                      OUTPUT                 PRINT-FILE

              PERFORM INPUT-A-RECORD

              PERFORM PRODUCE-TIME-LISTING
                  UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"

              WRITE PRINT-LINE
                  FROM EMPLOYEE-TOTAL-LINE

              CLOSE   CARD-FILE
                      PRINT-FILE
              STOP RUN

          INPUT-A-RECORD.

              READ CARD-FILE
                  INTO WORKING-TIME-CARD
                  AT END
                      MOVE "YES" TO END-OF-CARDS-SWITCH
```

شكل (١٩ - ١٢) تكملة

```
       PRODUCE-TIME-LISTING.

      *          INCREMENT COUNTER EACH TIME AN EMPLOYEE IS PROCESSED

           ADD 2 TO NUMBER-OF-EMPLOYEES

      *          MOVE DATA FROM INPUT CARD TO AREA WHERE LINE IS BUILT

           MOVE WORKING-NAME           TO LISTING-NAME
           MOVE WORKING-REGULAR-HOURS  TO LISTING-REGULAR-HOURS
           MOVE WORKING-OVERTIME-HOURS TO LISTING-OVERTIME-HOURS

      *          CALCULATE TOTAL HOURS BY ADDING REGULAR, OVERTIME
      *          NOTE THAT RESULT IS PLACED IN TIME-LISTING-LINE.

           ADD WORKING-REGULAR-HOURS WORKING-REGULAR-HOURS
               GIVING LISTING-TOTAL-HOURS

           WRITE PRINT-LINE
               FROM TIME-LISTING-LINE
```

شكل (٩ - ١٢) تكملة

```
ABEL FRED                  40     03     080
ABEL FRED                  40     03     080
ABEL FRED                  40     03     080
ABEL FRED                  40     03     080
.
.
.
```

شكل (٩ - ١٣)

```
DEBUG-ITEM = 000099 NUMBER-OF-EMPLOYEES                          000
INPUT-A-RECORD ,PRODUCE-TIME-LISTING ,
DEBUG-ITEM = 000129 NUMBER-OF-EMPLOYEES                          002
DEBUG-ITEM = 000134 WORKING-REGULAR-HOURS                        40
DEBUG-ITEM = 000135 WORKING-OVERTIME-HOURS                       03
DEBUG-ITEM = 000140 WORKING-REGULAR-HOURS                        40
DEBUG-ITEM = 000140 LISTING-TOTAL-HOURS                          080
PRODUCE-TIME-LISTING ,
DEBUG-ITEM = 000129 NUMBER-OF-EMPLOYEES                          004
DEBUG-ITEM = 000134 WORKING-REGULAR-HOURS                        40
DEBUG-ITEM = 000135 WORKING-OVERTIME-HOURS                       03
DEBUG-ITEM = 000140 WORKING-REGULAR-HOURS                        40
DEBUG-ITEM = 000140 LISTING-TOTAL-HOURS                          080
PRODUCE-TIME-LISTING ,
DEBUG-ITEM = 000129 NUMBER-OF-EMPLOYEES                          006
```

شكل (٩ - ١٤)

```
DEBUG-ITEM = 000134 WORKING-REGULAR-HOURS          40
DEBUG-ITEM = 000135 WORKING-OVERTIME-HOURS         03
DEBUG-ITEM = 000140 WORKING-REGULAR-HOURS          40
DEBUG-ITEM = 000140 LISTING-TOTAL-HOURS           080
.
.
.
```

شكل ( ٩ - ١٤ ) تكملة

إجراء تصحيح توضيحي :

١ - اختبر مخرجات الاختبار : يبدو أن أول سجل مدخلات تمت طباعته مرات ومرات .

٢ - اختبر مخرجات التصحيح ،التي تشمل READY TRACE ، وتعرض DEBUG - ITEM على ALL REFERENCES OF NUMBER - OF - EMPLOYEES, WORKING - REGULAR - HOURS, WORKING - OVER TIME-HOURS, LISTING - TOTAL - HOURS إذا جردنا مخرجات TRACE .. فإننا نجد ما يلي :

```
INPUT-A-RECORD, PRODUCE-TIME-LISTING,
PRODUCE-TIME-LISTING,
PRODUCE-TIME-LISTING,
...
```

من الواضح أن PRODUCE - TIME - LISTING ينفذ مرات ومرات مرة أخرى .

٣ - اختبار مخرجات DEBUG - ITEM لكل تنفيذ لـ .. PRODUCE - TIME - LISTING يظهر أن WORKING - OVERTIME - HOURS, WORKING - REGULAR - HOURS لا يتغيران ؛ تأكيدا لتكرار تنفيذ نفس السجل مرات ومرات .

٤ - بفحص البرنامج .. نجد أن السطرين ١٠٦ ، ١٠٧ يحتويان على :

```
PERFORM PRODUCE-TIME-LISTING
    UNTIL END-OF-CARDS-SWITCH IS EQUAL TO "YES"
```

٥ - بفحص الأسطر من ١٢٥ الى ١٤٦ .. نجد أن PRODUCE - TIME - LISTING لا يغير END - OF - CARDS - SWITCH ، ولا ينفذ PERFORM، أى مقطع مطلوب تنفيذه ،إيجازا،.، لا توجد طريقة لتنفيذ PERFORM ... UN-TIL حتى النهاية . من الواضح كذلك أن PRODUCE - TIME - LISTING قام بتشغيل نفس السجل مرات ومرات ، حيث لا يقدم الجزء مدخلات سجل منطقى جديد .

٦ - يمكن تصحيح كل من المشكلتين بإضافة العبارة التالية :

```
PERFORM INPUT-A-RECORD
```

فى نهاية المقطع PRODUCE - TIME - LISTING .. تقوم INPUT - A - RECORD بوضع END - OF - CARD - SWITCH فى نهاية الملف ، وكذلك وضع سجل المدخلات المنطقى التالى للتنفيذ التالى لـ PRODUCE - TIME - LIST- ING .

# الفصل العاشر
# معالجة الجداول
# Table Handling

سبق التعرض لمفهوم المنظومة المكون من عناصر بيانات متشابهة ، أو الجدول ، في الأقسام ١٤ من الفصل الخامس و ١٢ من الفصل السابع (انظر المثالين : ٧ - ٣٦ و ٧ - ٣٧ ) . ونتابع الموضوع بالتفصيل في الفصل الحالي . يقتصر اعتبارنا على جداول ذات بعد واحد وجداول ذات بعدين ، ولن يجد القارئ مشاكل في التوسع في معالجة جداول ذات بعدين ؛ لتشمل جداول ذات ثلاثة أبعاد - أكبر بعد ميسر في الكوبل (انظر ملحق حـ لمعرفة أكبر بعد ميسر في كوبل 1985 النمطي ) .

## ١٠ - ١ جداول ذات بعد واحد : جزء الحدوث

صيغة تعريف جداول ثابتة الطول ، ولها بعد واحد هي :

level-number data-name OCCURS integer TIMES

يمكن استخدام جزء الحدوث OCCURS على أي مستوى ماعدا المستوى 01 . ولكي يكون المبرمج قادرا على تحديد رقم صحيح integer .. يجب أن يعرف حجم الجدول مسبقا . وحيث إن هذا ليس مؤكدا دائما .. فمن المعتاد حجز أماكن كافية؛ لتحتوي على أقصى حجم يمكن احتياجه للجدول طوال فترة استخدام البرنامج .

**مثال ١٠ - ١ :**

```
01  MONTHLY-SALES-TABLE.
    05  MONTHLY-SALES          PIC S9(7)V99          COMP-3
                               OCCURS 12 TIMES.
```

يوجد بالتأكيد اثنا عشر شهرا في السنة .

```
•   01  MONTHLY-SALES-INFORMATION.
        05  SALESPERSON-DATA       OCCURS 400 TIMES.
            10  SALES-ID           PIC X(4).
            10  SALES-NAME         PIC X(20).
            10  MONTHLY-SALES      PIC S9(5)V99        COMP-3.
            10  QUOTA-AMOUNT       PIC S9(5)V99        COMP-3.
```

تكرر هنا مجموعة عناصر SALESPERSON - DATA : يوجد 400 محتوى فى الجدول ، يحتوى كل منها على مجموعة من أربعة حقول جزئية على المستوى 10 . وعلى هذا ، يكون طول المحتوى 4+4+20+4 أى 32 بايت (انظر القسم 18 من الفصل الخامس ) ، ويكون الجدول نفسه MONTHLY - SALES - INFORMATION ذا طول 400x32 ؛ أى 12800 بايت يمثل حجم الجدول 400 تخمينا آمنا لاقصى عدد بائعين ، يجرى عليهم تشغيل أثناء فترة استخدام البرنامج .

يمكن تعريف الجداول ذات البعد الواحد (والجدول ذى الأبعاد الأخرى ) كجزء من سجل فى قسم الملفات من جزء البيانات.

## ١٠ - ٢ الدلائل

بمعرفة أن جزء OCCURS يعرف عديداً من محتويات الجدول ( بوصف متطابق ) ... كيف يخبر المبرمج المترجم بأى المحتويات هو المطلوب فى احدى عبارات جزء الإجراءات ؟ أبسط طريقة هى استخدام دليل subscript ، وهو رقم صحيح موجب positive integer (فى صوره ثابت عددى أو عنصر بيانات عددى) ، يكتب بين قوسين بعد اسم الجدول .

**مثال ١٠ - ٢ :**

تستخدم الشفرة التالية أربعة أدلة فى تجميع محتويات جدول .

```
01  QUARTERLY-SALES.
    05  QUARTER-TOTAL          PIC S9(7)V99         COMP-3
                               OCCURS 4 TIMES.
. . . . . . . . . . . . . . . . . . .
PROCEDURE DIVISION.
. . . . . . . . . . . . . . . . . . .
    ADD QUARTER-TOTAL (1)
        QUARTER-TOTAL (2)
        QUARTER-TOTAL (3)
        QUARTER-TOTAL (4)
                               GIVING YEARLY-TOTAL
```

تظهر المنفعة الحقيقة للجداول والأدلة من الحقيقة بأنه يمكن استخدام عنصر البيانات الصحيح المتغير كدليل ، وتحدد القيمة الحالية للعنصر أى محتوى فى الجدول ، مطلوب إجراء تشغيل عليه . ومن المعتاد جدا استخدام عنصر بيانات كدليل ، يتم التحكم فيه عن طريق عبارة PERFORM ... VARYING .

**مثال ١٠ - ٣ :**

```
01  TOTAL-AREAS.
    05  LAST-YEARS-TOTAL     PIC S9(9)V99    COMP-3.
    05  THIS-YEARS-TOTAL     PIC S9(9)V99    COMP-3.
. . . . . . . . . . . . . . . . . . . . . .
01  SUBSCRIPT-AREAS.
    05  MONTH                PIC S9(2)       COMP SYNC.
. . . . . . . . . . . . . . . . . . . . . .
01  SALES-INFORMATION.
    05  MONTHLY-SALES    OCCURS 12 TIMES.
        10  LAST-YEAR        PIC S9(7)V99    COMP-3.
        10  THIS-YEAR        PIC S9(7)V99    COMP-3.
. . . . . . . . . . . . . . . . . . . . . .
    MOVE ZERO TO LAST-YEARS-TOTAL
                 THIS-YEARS-TOTAL
    PERFORM CALCULATE-YEARLY-TOTALS
        VARYING MONTH FROM 1 BY 1
            UNTIL MONTH GREATER THAN 12
. . . . . . . . . . . . . . . . . . . . . .
CALCULATE-YEARLY-TOTALS.
    ADD LAST-YEAR (MONTH) TO LAST-YEARS-TOTAL
    ADD THIS-YEAR (MONTH)  TO THIS-YEARS-TOTAL
```

هنا ، من جدول به اثنتا عشر قيمة لمجموعة العناصر MONTHLY - SALES ، وهى اثنتا عشر قيمة للعنصر الفردى LAST-YEAR ، واثنتا عشر قيمة للعنصر الفردى THIS - YEAR ، تجمع باستخدام دليل متغير MONTH .

فى كوبل IBM OS /VS .. الطريقة الأكثر كفاءة فى تعريف عنصر بيانات عددى لاستخدامه كدليل ، هى PIC S9 (n) COMP SYNC .. الدلائل مطلوبة لكل عناصر البيانات عند ، أو تابعة لـ ، جزء OCCURS .

## ١٠ - ٣ معالجة جداول ذات بعد واحد

### تحميل (وضع قيم ابتدائية) ، جدول بأصفار

يتكرر طلب وضع قيم ابتدائية لكل محتويات الجدول بنفس القيمة ، ولتكن صفر . ونظرا لعدم إمكانية استخدام جزء VALUE مع عناصر البيانات التى تكون جزءا من جدول .. فإن عبارة MOVE وسيلة بسيطة لوضع الأصفار فى الجدول.

**مثال ١٠ - ٤ :**

لتعريف جدول به 50 عنصر كما يلى :

```
01  SALES-FIGURES-BY-STATE.
    05  STATE-SALES          OCCURS 50 TIMES.
    10  STATE-ID             PIC X(2).
    10  SALES-AMOUNT         PIC S9(5)V99    COMP-3.
```

فإننا نحدد طريقة غير صحيحة ، والطريقة الصحيحة ، لوضع أصفار فى الجدول .

- MOVE ZEROS TO SALES-FIGURES-BY-STATE

عبارة صحيحة .. طالما أن SALES - FIGURES - BY - STATE ليست عنصر جدول ( فهى ليست موصوفة كجزء OCCURS ، وليست تابعة لجزء OCCURS ) ، وبالتالى .. فلا تتطلب دليلا . وحيث إنها مجموعة عناصر لها الشكل PIC X DISPLAY .. فإن أصفاراً إلى محتويات الجدول ، وبصفة خاصة ، فى 50 حدث لـ SALES - AMOUNT ، وهى عنصر من نوع COMP - 3 . وعلى هذا .. يوجد خطأ منطقى .

```
●       PERFORM ZERO-SALES-AMOUNT
            VARYING STATE-SALES-SUB
            FROM 1 BY 1
            UNTIL STATE-SALES-SUB GREATER THAN 50
    . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    ZERO-SALES-AMOUNT.
        MOVE ZERO TO SALES-AMOUNT (STATE-SALES-SUB)
        MOVE SPACES TO STATE-ID (STATE-SALES-SUB)
```

الإجراء الصحيح هو تعريف دليل (STATE - SALES - SUB) ، حيث يمكن توضيح الحقول الجزئية للجدول فى مقطع PERFORM .

مثال ١٠ - ٥ :

```
01  EMPLOYEE-DATA-TABLE.
    05  EMPLOYEE-INFORMATION      OCCURS 500 TIMES.
        10  SOCIAL-SEC-NUMBER     PIC 9(9).
        10  EMPLOYEE-SALARY       PIC S9(3)V99.
```

نظراً لأن كل الحقول من نوع DISPLAY .. فيمكننا أن نضع أصفاراً فى كل محتويات الجدول بعبارة واحدة ، وهى :

```
MOVE ZEROS TO EMPLOYEE-DATA-TABLE
```

## التحميل بجزء VALUE

إذا كان من الواجب وضع قيمة ابتدائية خاصة بكل عنصر من عناصر الجدول .. فإنه يمكن استخدام جزء VALUE فى إعداد مجموعه من عناصر البيانات التى يمكن بعد ذلك إعادة تعريفها REDEFINES كجدول . (الدوران ضرورى لأن جزء VALUE لا يمكن تطبيقه مباشرة على عناصر الجدول) . يمكن استخدام جزء VALUE بهذه الطريقة مع عناصر مخزن العمل.

مثال ١٠ - ٦ :

```
01  VALID-STATUS-CODES-TABLE.
    05  VALID-STATUS-VALUES.
        10  FILLER            PIC X(3)    VALUE "A07".
        10  FILLER            PIC X(3)    VALUE "A2K".
        10  FILLER            PIC X(3)    VALUE "B03".
        10  FILLER            PIC X(3)    VALUE "C99".
    05  STATUS-CODE           REDEFINES VALID-STATUS-VALUES
                              PIC X(3)
                              OCCURS 4 TIMES.
```

يحتوى VALID - STATUS - VALUES على أربعة عناصر فردية ، لكل منها PIC X (3) واستخدم جزء VALUE فى وضع قيم ابتدائية لها .

لاحظ أننا لا نهتم بإعطاء أسماء لعناصر البيانات هذه ، ولكن نستخدم الكلمة المحجوزة FILLER بدلا من ذلك . وعند ذلك.. يعاد تعريف VALID - STATUS - VALUES كجدول ، STATUS - CODE ، بطريقة تجعل العناصر المعرفة بأنها PIC X (3) فى STATUS - CODE والعناصر الأربعة المعرفة بأنها PIC X (3) فى VALID - STATUS - VALUES تكون هى نفسها (أى إنها تحتل نفس مواقع الذاكرة تماما) . وهذا يعنى أن STATUS - CODE (1) له قيمة بداية A07 و STATUS - CODE (2) له قيمة بداية A2K ... وهكذا . وتشير العبارات التى تجرى تشغيلا بالفعل على الجدول ، بالطبع ، إلى STATUS - CODE ، وليس إلى VALID - STATUS - VALUES .

**مثال ١٠ – ٧ :**

استخدم مثال ١٠ – ٦ جزء VALUE فى وضع قيم ابتدائية فى جدول به عناصر فردية ، والآن نضع قيما ابتدائية فى جدول به مجموعات عناصر .

```
01  SHIPPING-INSTRUCTIONS-TABLE.
    05  SHIPPING-CODES-VALUES.
        10  FILLER                    PIC X(2)     VALUE "A3".
        10  FILLER                    PIC X(20)    VALUE "U.S. MAIL".
        10  FILLER                    PIC X(2)     VALUE "A8".
        10  FILLER                    PIC X(20)    VALUE "UPS".
        10  FILLER                    PIC X(2)     VALUE "C3".
        10  FILLER                    PIC X(20)    VALUE "FEDERAL EXPRESS".
    05  CODES-AND-INSTRUCTIONS        REDEFINES SHIPPING-CODES-VALUES
                                      OCCURS 3 TIMES.
        10  SHIPPING-CODE             PIC X(2).
        10  SHIPPING-INSTRUCTION      PIC X(20).
```

مرة أخرى .. تعرف عناصر الجدول كعناصر فردية أولا باستخدام اسم FILLER ، وأجزاء VALUE, USAGE, PIC المناسبة . لاحظ أن SHIPPING - CODES - VALUES تحتوى على عنصر له PIC X (2) ، يتبعه عنصر له PIC X (20) ، يتبعه عنصر ثان له PIC X (1) وهكذا . ويقوم CODES - AND - INSTRUCTIONS بإعادة تعريف SHIPPING - CODES - VALUES كجدول به ٣ عناصر ؛ حيث يحتوى كل عنصر SHIPPING - CODE له PIC X (2)، يليه SHIPPING - INSTRUCTION له PIC X (20) . بسبب أجزاء VALUE .. فإن SHIPPING - CODE (1) له قيمة بداية "A3" ، وتكون لـ SHIPPING - INSTRUCTION (1) قيمة بداية "U.S. MAIL" ، وتكون لـ SHIPPING - CODE (2) قيمة بداية "A8" كما أن SHIPPING - INSTRUCTION (2) تكون له قيمة بداية "USP" .. وهكذا . يطبق مثال (١٠ – ٢٠ ) هذا الأسلوب على جدول به عناصر عددية .

## التحميل بعبارة READ

هناك عيبان من وضع القيم الابتدائية فى جداول مخزن العمل عن طريق إعادة تعريف العناصر بعناصر لها جزء VALUE: (١) مع ازدياد حجم الجداول وتعقيدها .. تصبح هذه الطريقة مزعجة جدا وتتطلب شفرة كثيرة فى جزء البيانات (٢)

اذا كان لابد من تغيير القيم الابتدائية لعناصر الجدول .. فيجب تعديل برنامج الكوبل نفسه وإعادة ترجمته . وهناك طريقة أخرى فعالة لتجنب هذه الصعوبات ، وهى حفظ قيم بداية الجدول فى ملف قرص تتابعى أو ملف شريط تتابعى ، وعندما يبدأ تنفيذ البرنامج .. يمكنه أن يقرأ READ القيم الابتدائية فى المناطق المناسبة من الجدول . فإذا كانت هناك حاجة لتغيير قيم البداية للجدول .. فيمكن عمل ذلك بسهولة عن طريق تعديل الملف .

**مثال ١٠ - ٨ :**

جدول رموز الحالة الخاص بمثال ( ١٠ - ٦ ) من ملف بالطريقة التالية :

```
FD  STATUS-CODE-FILE...
01  STATUS-CODE-RECORD.
    05  STATUS-CODE-VALUE        PIC  X(3).
    . . . . . . . . . . . . . . . . . . . . .
WORKING-STORAGE SECTION.
    . . . . . . . . . . . . . . . . . . . . .
01  VALID-STATUS-CODES-TABLE.
    05  STATUS-CODE              PIC  X(3)    OCCURS 400 TIMES.
    05  NUMBER-OF-CODES          PIC  S9(3)   COMP SYNC.
    . . . . . . . . . . . . . . . . . . . . .
PROCEDURE DIVISION.
    OPEN  INPUT  STATUS-CODE-FILE
    MOVE "NO" TO END-STATUS-FILE-SW
    PERFORM LOAD-STATUS-TABLE
        VARYING NUMBER-OF-CODES FROM 1 BY 1
        UNTIL       END-STATUS-FILE-SW EQUAL "YES"
              OR    NUMBER-OF-CODES GREATER THAN 400
    IF NUMBER-OF-CODES GREATER THAN 400
        MOVE 400 TO NUMBER-OF-CODES
        DISPLAY "TABLE FILE TOO LARGE--MODIFY, RECOMPILE COBOL TABLE"
            UPON OPERATOR-MESSAGE-DEVICE
    ELSE
        SUBTRACT 1 FROM NUMBER-OF-CODES
    .
    CLOSE STATUS-CODE-FILE
    . . . . . . . . . . . . . . . . . . . . . . . .
LOAD-STATUS-TABLE.
    READ STATUS-CODE-FILE
        AT END
            MOVE "YES" TO END-STATUS-FILE-SW
    .
    IF END-STATUS-FILE-SW EQUAL "NO"
        MOVE STATUS-CODE-VALUE TO STATUS-CODE (NUMBER-OF-CODES)
```

لاحظ كيف استخدم NUMBER - OF - CODES فى عد عدد عناصر الجدول التى أدخلت فعلا من الملف ، و لاحظ أن دورة PERFORM ... VARYING التى تدخل قيم شفرة الحالة تتوقف عند نهاية الملف ، أو إذا تعدى NUMBER - OF - CODES حجم الجدول ، كما هو موضح فى جزء OCCURS (400 فى هذه الحالة ) . عبارة IF تلى .. PERFORM VARYING تختبر عند ذلك NUMBER - OF - CODES ، وإذا تعدى أقصى حجم للجدول ، تعرض رسالة لمشغل الكمبيوتر ، التى تحدد أن جزء OCCURS يجب أن يتغير وتعاد ترجمة البرنامج . لاحظ إنه إذا حدثت نهاية الملف .. فإن آخر قيمة ) NUMBER - OF - CODES تناظر نهاية الملف ، وليست لـ STATUS - CODE - VALUE . وعلى هذا .. يجب أن يقل NUMBER - OF - CODES بمقدار 1، بعد توقف PERFORM ... VARYING عند نهاية الملف . وفى أى حالة .. يحدد NUMBER - OF - CODES – بطريقة صحيحة – عدد عناصر الجدول المحملة لأى مرة من مرات تنفيذ برنامج معين .

إن حفظ عداد بعدد العناصر النشطة فى الجدول شائع الاستخدام جدا كوسيله فى معالجة الجداول .

## تجميع عناصر الجدول

بالإضافة إلى مثال ١٠ – ٣ فإننا نضرب مثالاً آخر .

**مثال ١٠ – ٩ :**

أفرض أنه مطلوب تجميع مجموع رواتب العاملين فى جدول مثال ( ١٠ – ٥ ) .

```
01  EMPLOYEE-DATA-TABLE.
    05  EMPLOYEE-INFORMATION          OCCURS 500 TIMES.
        10  SOCIAL-SEC-NUMBER         PIC 9(9).
        10  EMPLOYEE-SALARY           PIC S9(3)V99    COMP-3.
    05  NUMBER-ACTIVE-EMPLOYEES       PIC S9(3)       COMP.
    . . . . . . . . . . . . . . . . . . . . . . . . . .
    MOVE ZERO TO TOTAL-SALARIES
    PERFORM TOTAL-EMPLOYEE-SALARIES
        VARYING EMPLOYEE-SUB
        FROM 1 BY 1
        UNTIL EMPLOYEE-SUB GREATER THAN NUMBER-ACTIVE-EMPLOYEES
    . . . . . . . . . . . . . . . . . . . . . . . . . .
TOTAL-EMPLOYEE-SALARIES.
    ADD EMPLOYEE-SALARY (EMPLOYEE-SUB) TO TOTAL-SALARIES
```

لم نجمع هنا كل 500 عنصر ، فبالرغم من أن الجدول معرف ليحتوى 500 عامل كحد أقصى ، إلا أنه قد لا يحتوى إلا على بيانات لعدد أقل من 500 عامل . يستخدم عنصر البيانات NUMBER - ACTIVE - EMPLOYEES ( وهو بنفسه لا يعد جزءا من الجدول ) فى حفظ تتبع عدد العاملين الممثلين فعلا فى الجدول . ويفترض أن الاجزاء التى تضيف عاملين جدد الى الجدول ، أو تحذف عاملين منه تظل محتفظة بهذا العدد كذلك . يصبح NUMBER - ACTIVE - EMPLOYEES قيمة إيقاف عبارة PERFORM التى تتحكم فى جمع رواتب العاملين .

## فحص الجدول باستخدام PERFORM

فى عمليه الفحص look - up أو البحث search .. تقارن عناصر الجدول مع قيمة معينة حتى يوجد العنصر المتوافق مع هذه القيمة ، أو حتى يتقرر عدم وجود مثل هذه القيمة فى الجدول . وأبسط خوارزمى للبحث فى جدول ، هو البحث التتابعى se-quential أو الخطى linear ؛ حيث يفحص فيه أول عنصر ... فالثانى ، فالثالث وهكذا ، حتى يتم الوصول الى العنصر المطلوب، أو ينتهى الجدول . توفر معظم صيغ الكوبل عبارة بحث SEARCH ( انظر القسم الثامن من هذا الفصل ) لبرمجة البحث التتابعى ، ولكنه يمكن كذلك عمل البرمجة باستخدام دورة PERFORM .

**مثال ١٠ - ١٠ :**

بمعرفة الجدول المعطى فى مثال (١٠ - ٦)، وقيمة معروفة فى INPUT - STATUS - CODE PIC X (3) : فالمطلوب تحديد ما اذا كان INPUT - STATUS - CODE صحيحا، وذلك بفحص الجدول VALID- STATUS - CODE - TABLE.

```
01  VALID-STATUS-CODES-TABLE.
    05  STATUS-CODE              PIC X(3)
                                 OCCURS 40 TIMES.
. . . . . . . . . . . . . . . . . . . . . . . . .
CHECK-STATUS-CODE.
    PERFORM LOOK-UP-STATUS
    IF VALID-STATUS-SW EQUAL "YES"
    . . . (process valid status code)
    ELSE
    . . . (handle invalid status code)
. . . . . . . . . . . . . . . . . . . . . . . .
LOOK-UP-STATUS.
    MOVE "NO" TO VALID-STATUS-SW
    PERFORM CHECK-ENTRY
        VARYING STATUS-CODE-SUBSCRIPT FROM 1 BY 1
        UNTIL     STATUS-CODE-SUBSCRIPT GREATER THAN 40
              OR VALID-STATUS-SW EQUAL "YES"

CHECK-ENTRY.
    IF STATUS-CODE (STATUS-CODE-SUBSCRIPT) EQUAL INPUT-STATUS-CODE
        MOVE "YES" TO VALID-STATUS-SW
```

يضع LOOK - UP - STATUS القيمة "NO" فى VALID - STATUS - SW أولا . وينفذ CHECK - ENTRY بعد ذلك ، مع تغيير STATUS - CODE - SUBSCRIPT ( المعرف على أنه PIC S9(5) COMP SYNC ) من 1 حتى عدد عناصر الجدول ؛ فإذا تساوى أحد عناصر الجدول المناظر للدليل STATUS - CODE - SUBSCRIPT – فى أى لحظة – مع INPUT - STATUS - CODE .. فتوضع YES فى VALID - STATUS - SW .

وعلى هذا .. يتوقف التنفيذ لعبارة PERFORM ... UNTIL . فاذا لم يتساو أى عنصر من عناصر الجدول مع INPUT - STATUS - CODE .. فإن القيمة "NO" تظل موجودة فى VALID - STATUS - SW .

لاحظ إنه اذا تساوى العنصر السادس (مثلا) مع INPUT - STATUS - CODE .. تتوقف عبارة PERFORM ، مع مساواة STATUS - CODE - SUBSCRIPT بالرقم ٧ (شكل ٧ - ١٤) . و يمكن معالجة هذا القصور كما يلى :

```
LOOK-UP-STATUS.
    MOVE 1 TO STATUS-CODE-SUBSCRIPT
    MOVE "YES" TO VALID-STATUS-SW
    PERFORM CHECK-ENTRY
        UNTIL VALID-STATUS-SW EQUAL "NO"
            OR STATUS-CODE (STATUS-CODE-SUBSCRIPT) EQUAL
                    INPUT-STATUS-CODE

CHECK-ENTRY.
IF STATUS-CODE-SUBSCRIPT EQUAL 40
    MOVE "NO" TO VALID-STATUS-SW
ELSE
    ADD 1 TO STATUS-CODE-SUBSCRIPT
```

والآن إذا أشار STATUS - CODE - SUBSCRIPT إلى عنصر بأنه يساوى INPUT - STATUS - CEOD .. فإن شرط UNTIL يتحقق وينتهى تنفيذ PERFORM فورا دون زيادة STATUS - CODE - SUBSCRIPT والذى يشير إلى العنصر المطلوب فى الجدول .

**مثال ١٠ - ١١ :**

هناك سبب منتشر لاستخدام البحث فى الجدول ، وهو فك شفرة decode حقول مدخلات ؛ لطباعتها فى تقارير ، ونوضح ذلك بجدول يشبه جدول مثال ( ٧ - ١٠ ) .

```
01  SHIPPING-INSTRUCTIONS-TABLE.
    05  CODES-AND-INSTRUCTIONS        OCCURS 18 TIMES.
        10  SHIPPING-CODE             PIC X(2).
        10  SHIPPING-INSTRUCTION      PIC X(30).
. . . . . . . . . . . . . . . . . . . . . . . . . . .
    MOVE "YES" TO FOUND-CODE-SW
    MOVE 1 TO SHIPPING-SUBSCRIPT
    PERFORM CHECK-ENTRY
        UNTIL       SHIPPING-CODE (SHIPPING-SUBSCRIPT) EQUAL INPUT-CODE
                OR FOUND-CODE-SW  EQUAL "NO"
    IF FOUND-CODE-SW EQUAL "YES"
        MOVE SHIPPING-INSTRUCTION (SHIPPING-SUBSCRIPT) TO
            PRINTED-SHIPPING-MESSAGE
    ELSE
        MOVE "*** INVALID SHIPPING CODE ***" TO
            PRINTED-SHIPPING-MESSAGE

. . . . . . . . . . . . . . . . . . . . . . . . . . .
CHECK-ENTRY.
    IF SHIPPING-SUBSCRIPT EQUAL 18
        MOVE "NO" TO FOUND-CODE-SW
    ELSE
        ADD 1 TO SHIPPING-SUBSCRIPT
```

إذا وجد INPUT - CODE فى SHIPPING - INSTRUCTION - TABLE . فإن SHIPPING - INSTRUC-TION المناظر لـ SHIPPING - CODE – الذى يتفق مع INPUT - CODE – ينقل إلى منطقة طباعة المخرجات ؛ فإذا لم يوجد INPUT - CODE فى الجدول .. فتطبع رسالة خطأ فى التقرير . وتسمح هذه الطريقة بأن تحتوى سجلات المدخلات على رمز من 2 بايت فقط ، مقدمة على ذلك تعليمات الشحن المكون من 30 بايت فى تقرير الطباعه .

## ١٠ - ٤ جداول ذات بعدين

عندما تعرف عناصر تابعة لأحد عناصر OCCURS لها جزء OCCURS أيضا .. فإننا نقول إن لهذا الجدول بعدين two - dimensions .

**مثال ١٠ – ١٢ :**

```
01  FREIGHT-CHARGES-TABLE.
    05  WEIGHT-RANGE                OCCURS  3 TIMES.
        10  WEIGHT-LIMIT            PIC  S9(3)              COMP-3.
        10  DESTINATION             OCCURS  3 TIMES.
            15  FREIGHT-CHARGE      PIC  S9(3)V99           COMP-3.
```

WEIGHT - RANGE هو جدول به ٣ عناصر ، ويحتوى كل عنصر على الحقول DESTINATION , WEIGHT - LIMIT ... إلا أن DESTINATION نفسه عبارة عن جدول به ٣ عناصر (يحتوى كل منها على قيمة FREIGHT - CHARGE ) . وعلى هذا .. فالجدول التابع DESTINATION يحتوى على ٣ × ٣ = ٩ عناصر ، انظر شكل (١٠ – ١) ، لاحظ أن عدد الدلائل المطلوب للإشارة الى أحد عناصر الجدول يساوى عدد أجزاء OCCURS المستخدمة مع العنصر . يشير أول دليل الى أعلى مستوى لجزء OCCURS ، ويشير الدليل الثانى للمستوى التالى له ... وهكذا . وبالنسبة لهذا الجدول الخـاص الذى له بعدان .. فـإن الدليلين لهما نفس مدى القيم ، ولن يكون هـذا صحيحا بالطبع لكـل الجـداول . فى شكـل (١٠ – ١) .. تفصل الدلائل المتعددة بواسطة فاصلة وفراغ ، الفاصلة اختيارية .

| | | | |
|---|---|---|---|
| WEIGHT-RANGE (1) | WEIGHT-LIMIT (1) | 2 bytes, S9(3) | COMP-3 |
| | FREIGHT-CHARGE (1, 1) | 3 bytes, S9(3)V99 | COMP-3 |
| | FREIGHT-CHARGE (1, 2) | 3 bytes, S9(3)V99 | COMP-3 |
| | FREIGHT-CHARGE (1, 3) | 3 bytes, S9(3)V99 | COMP-3 |
| WEIGHT-RANGE (2) | WEIGHT-LIMIT (2) | 2 bytes, S9(3) | COMP-3 |
| | FREIGHT-CHARGE (2, 1) | 3 bytes, S9(3)V99 | COMP-3 |
| | FREIGHT-CHARGE (2, 2) | 3 bytes, S9(3)V99 | COMP-3 |
| | FREIGHT-CHARGE (2, 3) | 3 bytes, S9(3)V99 | COMP-3 |
| WEIGHT-RANGE (3) | WEIGHT-LIMIT (3) | 2 bytes, S9(3) | COMP-3 |
| | FREIGHT-CHARGE (3, 1) | 3 bytes, S9(3)V99 | COMP-3 |
| | FREIGHT-CHARGE (3, 2) | 3 bytes, S9(3)V99 | COMP-3 |
| | FREIGHT-CHARGE (3, 3) | 3 bytes, S9(3)V99 | COMP-3 |

شكل ( ١٠ - ١ )

**مثال ١٠ - ١٣ :**

إذا كان السجل STUDENT - TRANSCRIPT – المعرف أدناه سجلات – منطقيا لملف الطلبة الرئيسى ، فما طول السجل المنطقى ؟

```
01  STUDENT-TRANSCRIPT.
    05  STUDENT-ID                PIC X(9).
    05  STUDENT-NAME              PIC X(25).
    05  SEMESTERS-ATTENDED        PIC S9          COMP-3.
    05  SEMESTER-INFORMATION      OCCURS 8 TIMES.
        10  GRADE-POINT-AVERAGE   PIC S9V99       COMP-3.
        10  NUMBER-COURSES        PIC S9          COMP-3.
        10  COURSE-INFO           OCCURS 6 TIMES.
            15  COURSE-ID         PIC X(5).
            15  NUMBER-CREDITS    PIC S9V9        COMP-3.
            15  GRADE             PIC S9          COMP-3.
```

تعريف المقرر COURSE - ID طوله 5 بايت ، وعدد الساعات NUMBER - CREDITS ( القسم ١٨ – من الفصل الخامس ) طوله 2 بايت وطول الدرجة GRADE هو بايت واحد ، وعلى هذا .. فعنصر معلومات المقرر COURSE - INFO طوله ٨ بايت . وحيث إن COURSE - INFO OCCURS .. فإن محتويات الجدول COURSE - INFO يبلغ طولها 48 بايت. يحتوى عنصر SEMESTER - INFORMATION على محتويات الجدول COURSE - INFO ؛ مضافا اليها NUMBER - COURSES ( طوله بايت واحد ) وكذلك GRADE - POINT - AVERAGE ( طوله 2 بايت ) ، وعلى هذا .. يصبح طول SE-

عدد 51 بايت، وحيث إنه هناك 8 عناصر تتطلب محتويات الجدول SEMESTER - IN- MESTER - INFORMATION عدد 408 بايت . ويجب أن نجمع عليها مكانا للحقل لرقم تعريف الطالب STUDENT - ID (9 بايت)، ومكانا آخر لاسم الطالب STUDENT - NAME (25 بايت) ، ومكانا آخر لعدد الفصول الدراسية التى درسها SEMESTERS - ATTENDED ( واحد بايت ) ، ويصبح إجمالى طول السجل 443 بايت .

يجب ملاحظة أن تعريف STUDENT - TRANSCRIPT يفترض مسبقا تحميل الجدول ؛ بحيث يشير COURSE - INFO (i,j) إلى المقرر رقم j الذى درسه الطالب أثناء الفصل i خلال دراسته ( أى انه إذا كان الجدول معروضا على هيئة مصفوفة .. فنجد أنه يضغط فى اتجاه الركن العلوى الأيسر ) . يخزن عدد الفصول الدراسية التى درسها الطالب فى العداد SEMESTER - ATTENDED ، كما يخزن العداد NUMBER - COURSES (k) عدد المقررات التى درسها الطالب أثناء الفصل الدراسى k فى دراسته . تحدد هذه العدادات أيًا من مواقع ذاكرة COURSE - INFO البالغ عددها ٤٨ (٨ × ٦) بالضبط ، يحتوى على بيانات نشطة .

**مثال ١٠ – ١٤ :**

أحسب متوسط اداء الفصل الدراسى لطالب معلوماته موضوعه حاليا فى السجل المنطقى STUDENT - TRANSCRIPT الموجود فى مثال (١٠ – ١٣) .

```
MOVE ZERO TO GRADE-POINT-TOTAL
PERFORM ADD-UP-SEMESTER-AVERAGES
    VARYING SEMESTER-SUB FROM 1 BY 1
    UNTIL SEMESTER-SUB GREATER THAN SEMESTERS-ATTENDED
DIVIDE SEMESTERS-ATTENDED INTO GRADE-POINT-TOTAL
    GIVING AVERAGE-SEMESTER-GRADE
. . . . . . . . . . . . . . . . . . . . . . . . . . .
ADD-UP-SEMESTER-AVERAGES.
    ADD GRADE-POINT-AVERAGE (SEMESTER-SUB) TO GRADE-POINT-TOTAL
```

## ١٠ – ٥ معالجة جداول ذات بعدين

### تحميل أصفار

حيث إنه يستخدم دليلان ... فتستخدم عبارة ... PERFORM ... VARYING ... AFTER

مثال ١٠ - ١٥ :

يفترض فيما يلى أن كل محتويات الجدول الستة نشطة حاليا .

```
01  SALES-AMOUNTS.
    05  REGION-TOTALS                OCCURS 2 TIMES.
        10  DEPARTMENT-TOTALS        OCCURS 3 TIMES.
            15  SALES                PIC S9(5)V99   COMP.
. . . . . . . . . . . . . . . . . . . . . . . . .
01  SUBSCRIPTS.
    05  REGION-NUMBER                PIC S9(5)   COMP SYNC.
    05  DEPARTMENT-NUMBER            PIC S9(5)   COMP SYNC.
. . . . . . . . . . . . . . . . . . . . . . . . .
    PERFORM ZERO-SALES-AMOUNTS
        VARYING REGION-NUMBER FROM 1 BY 1
            UNTIL REGION-NUMBER GREATER THAN 2
        AFTER DEPARTMENT-NUMBER FROM 1 BY 1
            UNTIL DEPARTMENT-NUMBER GREATER THAN 3
. . . . . . . . . . . . . . . . . . . . . . . . .
ZERO-SALES-AMOUNTS.
    MOVE ZERO TO SALES (REGION-NUMBER DEPARTMENT-NUMBER)
```

## جمع المحتويات

مثال ١٠ - ١٦ :

احسب إجمالى الساعات حتى الآن للطالب فى مثال (١٠ - ١٣) .

```
. . . . . . . . . . . . . . . . . . . . . . . . .
05  TOTAL-CREDITS      PIC S9(3)V9      COMP-3.
. . . . . . . . . . . . . . . . . . . . . . . . .
MOVE ZERO TO TOTAL-CREDITS
PERFORM ACCUMULATE-TOTAL-CREDITS
    VARYING SEMESTER-NUMBER FROM 1 BY 1
        UNTIL SEMESTER-NUMBER GREATER THAN SEMESTERS-ATTENDED
    AFTER COURSE-NUMBER FROM 1 BY 1
        UNTIL COURSE-NUMBER GREATER THAN
            NUMBER-COURSES (SEMESTER-NUMBER)
. . . . . . . . . . . . . . . . . . . . . . . . .
ACCUMULATE-TOTAL-CREDITS.
    ADD NUMBER-CREDITS (SEMESTER-NUMBER COURSE-NUMBER)
        TO TOTAL-CREDITS
```

حيث تعرف الأدلة كما يلى :

```
01  SUBSCRIPTS.
    05  SEMESTER-NUMBER      PIC S9(5)  COMP SYNC.
    05  COURSE-NUMBER        PIC S9(5)  COMP SYNC.
```

لاحظ بصفة خاصة الأدلة الخاصة بالعنصر NUMBER - COURSES فى عبارة PERFORM . ونظرا لتكرار حدوث هذ العنصر ٨ مرات – كجزء من الجدول SEMESTER - INFORMATION – يجب أن تكون له أدلة . و فى كل مرة يزداد SEMESTER - NUMBER .. تتغير قيمة NUMBER - COURSES (SEMESTER - NUMBER) طبقا لهذه الزيادة.

**مثال ١٠ – ١٧ :**

فى مثال ( ١٠ – ١٣) ، ومثال (١٠ – ١٤) نجد أن محتويات GRADE - POINT - AVERAGE ، وهى جزء من جدول SEMSTER - INFORMATIOON سبق معاملتها كبيانات مدخلات . افرض أنه من الضرورى حساب GRADE -POINT - AVERAGE . من جدول COURSE - INFO : يمكن أن يحدث هذا ، لكل فصل دراسى ، بتجميع الدرجات مضروبة فى عدد الساعات لكل المقررات التى درست خلال الفصل الدراسى وقسمتها على إجمالى عدد الساعات التى درست خلال هذا الفصل الدراسى ، ويمكن أن تشمل شفرة الكوبل هيكلاً PERFORM ... VARYING متداخلاً ، كما هو مبين أدناه ، حيث الأدلة كما هى معرفة فى مثال (١٠ – ١٦) ، وحيث العناصر TOTAL - CREDITS , TOTAL - GRADE - POINTS هى عناصر عددية فى مخزن العمل .

```
    PERFORM CALCULATE-SEMESTER-AVERAGE
        VARYING SEMESTER-NUMBER FROM 1 BY 1
            UNTIL SEMESTER-NUMBER GREATER THAN SEMESTERS-ATTENDED
. . . . . . . . . . . . . . . . . . . . . . . . . . . . .
CALCULATE-SEMESTER-AVERAGE.
    MOVE ZERO TO TOTAL-GRADE-POINTS
                 TOTAL-CREDITS
    PERFORM TOTAL-SEMESTER-COURSES
        VARYING COURSE-NUMBER FROM 1 BY 1
            UNTIL COURSE-NUMBER GREATER THAN
                NUMBER-COURSES (SEMESTER-NUMBER)
    DIVIDE TOTAL-CREDITS INTO TOTAL-GRADE-POINTS
        GIVING GRADE-POINT-AVERAGE (SEMESTER-NUMBER)
    .
TOTAL-SEMESTER-COURSES.
    ADD NUMBER-CREDITS (SEMESTER-NUMBER COURSE-NUMBER)
        TO TOTAL-CREDITS
    COMPUTE TOTAL-GRADE-POINTS = TOTAL-GRADE-POINTS +
        GRADE (SEMESTER-NUMBER COURSE-NUMBER) *
            NUMBER-CREDITS (SEMESTER-NUMBER COURSE-NUMBER)
    .
```

## تحويل بيانات مدخلات الى دلائل

بالنسبة للجداول التى تعرضنا لها حتى الآن ، كانت الدلائل تنتج بواسطة ... PERFORM ... VARYING ، ولكن الحصول على قيم دلائل عن طريق تحويل بيانات مدخلات يكون مطلوبا فى بعض الأحيان.

أبسط تحويل هو عدم التحويل – وذلك عندما تكون الحالة بيانات المدخلات (طبقا لتصميمها) مناسبة كما هى .

**مثال ١٠ – ١٨ :**

افرض أن العامل فى إدخال البيانات .. ادخل حقول DISPLAY التالية :

```
FD  SALES-INPUT-FILE...
01  SALES-RECORD.
    05  REGION-ID          PIC 9(1).
    05  DEPARTMENT-ID      PIC 9(2).
    05  SALES-AMOUNT       PIC S9(4)V99.
```

ينتج عن استخدام DEPARTMENT - ID وREGION - ID كدلائل جدول لإجمالى المبيعات ؛ طبقا للمنطقة والقسم.

```
01  TOTAL-SALES-AMOUNTS-TABLE.
    05  REGION                      OCCURS 7 TIMES.
        10  DEPARTMENT              OCCURS 12 TIMES.
            15  TOTAL-SALES-AMOUNT  PIC S9(5)V99  COMP.
    05  REGION-LIMIT                PIC S9(2)     COMP SYNC  VALUE + 7.
    05  DEPARTMENT-LIMIT            PIC S9(2)     COMP SYNC  VALUE + 12.
```

يمكن أن تعمل الشفرة التالية ( يفترض أن TOTAL - SALES - AMOUNT قد سبق وضع أصفار فى محتوياتها ) :

```
    . . . . . . . . . . . . . . . . . . . . . . . . . .
    OPEN  INPUT  SALES-INPUT-FILE
    MOVE "NO" TO END-FILE-SW
    PERFORM GET-SALES-RECORD
    PERFORM ACCUMULATE-SALES-TOTALS
        UNTIL END-FILE-SW EQUAL "YES"
    CLOSE SALES-INPUT-FILE
    . . . . . . . . . . . . . . . . . . . . . . . . . .
GET-SALES-RECORD.
    READ SALES-INPUT-FILE
        AT END
            MOVE "YES" TO END-FILE-SW
    .
ACCUMULATE-SALES-TOTALS.
    IF REGION-ID NOT NUMERIC OR DEPARTMENT-ID NOT NUMERIC OR
        SALES-AMOUNT NOT NUMERIC
            PERFORM INVALID-NUMBER-ROUTINE
    ELSE IF REGION-ID LESS THAN 1 OR GREATER THAN REGION-LIMIT
        PERFORM REGION-OUT-OF-RANGE-ROUTINE
    ELSE IF DEPARTMENT-ID LESS THAN 1 OR GREATER THAN
        DEPARTMENT-LIMIT
        PERFORM DEPARTMENT-OUT-OF-RANGE-ROUTINE
    ELSE
        ADD SALES-AMOUNT TO
            TOTAL-SALES-AMOUNT (REGION-ID DEPARTMENT-ID)
    .
    PERFORM GET-SALES-RECORD
```

لاحظ التأكد من عناصر المدخلات المستخدمة كدلائل ، لاتقدم معظم صيغ الكوبل أى تحذير تلقائى ، وذلك عندما تقع إشارة بدلائل خارج الجدول .

يمكن أن يتحقق التحويل الفعلى لبيانات المدخلات باستخدام عبارة IF خطية ، على أن يكون عدد القيم صغيرا نسبيا ، وإلا تحقق ذلك بواسطة فحص الجدول . يوضح مثال (١٠ – ١٩) كلاً من الطريقتين .

**مثال ١٠ – ١٩ :**

أعد حل مثال (١٠ – ١٨) ، إذا كان حجم الجدول 3x15 ( بدلا من 7x12 ) . وإذا كانت الحقول الحرفية عددية معطاة على النحو التالى :

```
01  SALES-RECORD.
    05  REGION-ID          PIC X(3).
    05  DEPARTMENT-ID      PIC X(5).
    05  SALES-AMOUNT       PIC S9(4)V99.
```

يمكننا استخدام جدول ذى بعد واحد فى تحويل DEPARTMENT - ID الى الدليل DEPARTMENT - NUMBER .

```
01  DEPARTMENT-ID-TABLE.
    05  VALID-DEPARTMENT-ID     OCCURS 15 TIMES
                                PIC X(5).
```

يتحول أول عنصر فى جدول DEPARTMENT - ID - TABLE الى القيمه 1 من DEPARTMENT - NUMBER، والعنصر الثانى الى القيمة 2 وهكذا . وعلى هذا .. يمكننا أن نحصل على DEPARTMENT - NUMBER بالبحث عن DEPARTMENT - ID فى الجدول . ويبين هذا البحث ، مع هيكل عبارة IF الخطية التى تحول REGION - ID الى دليل REGION - NUNBER ، فى جزء التركيم المراجع للمشكلة .

```
ACCUMULATE-SALES-TOTAL.
    MOVE "NO" TO ERROR-SW
    IF REGION-ID EQUAL "XYZ"
        MOVE 1 TO REGION-NUMBER
    ELSE IF REGION-ID EQUAL "ABC"
        MOVE 2 TO REGION-NUMBER
    ELSE IF REGION-ID EQUAL "GHF"
        MOVE 3 TO REGION-NUMBER
    ELSE
        MOVE "YES" TO ERROR-SW
    .
    MOVE 1 TO DEPARTMENT-NUMBER
    PERFORM LOOKUP-DEPARTMENT
        UNTIL ERROR-SW EQUAL "YES" OR
                DEPARTMENT-ID EQUAL
                    VALID-DEPARTMENT-ID (DEPARTMENT-NUMBER)
    IF SALES-AMOUNT NOT NUMERIC
        MOVE "YES" TO ERROR-SW
```

```
    IF ERROR-SW EQUAL "YES"
          PERFORM INVALID-INPUT-ROUTINE
    ELSE
       ADD SALES-AMOUNT TO
          TOTAL-SALES-AMOUNT (REGION-NUMBER DEPARTMENT-NUMBER)
    .
    PERFORM GET-SALES-RECORD
    .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
LOOKUP-DEPARTMENT.
    IF DEPARTMENT-NUMBER EQUAL DEPARTMENT-LIMIT
       MOVE "YES" TO ERROR-SW
    ELSE
       ADD 1 TO DEPARTMENT-NUMBER
```

فى المشاكل الأكثر واقعية .. يجب ألا تكتب قيم REGION - ID كثوابت ، وإنما تكتب كعناصر بيانات فى مخزن العمل ، لها قيم VALUE المطلوبة ، أى إن :

```
05  REGION-ONE-ID     PIC X(3)     VALUE "XYZ".
```

إذا لم تميز REGION - ID .. يعد مفتاح للخطأ ، وبالمثل .. إذا لم يوجد DEPARTMENT - ID فى ID الموجودة فى الجدول .. يعد نفس مفتاح الخطأ ، ويعد مفتاح الخطأ كذلك إذا كان SALES - AMOUNT غير عددى .

## التحميل بجزء VALUE

كما هو الحال مع الجداول ذات البعد الواحد ( انظر القسم ٣ من هذا الفصل) .. فإنه من الممكن وضع قيم ابتدائية لجدول ذى بعدين ، وذلك بإعادة تعريف منطقة يتفق تكوينها مع الجدول مرة أخرى . لا يوصى بهذه الطريقة إذا كان من المتوقع أن تجرى تغييرات على الجدول خلال فترة استخدامه .

**مثال ١٠ - ٢٠ :**

فيما يلى وضع قيم ابتدائية لجدول مثال (١٠ - ١٢) .

```
01  FREIGHT-CHARGES-AREA.
    05  FREIGHT-CHARGE-VALUES.
10  FILLER.
    15  FILLER      PIC S9(3)      COMP-3     VALUE +50.
    15  FILLER      PIC S9(3)V99   COMP-3     VALUE +1.50.
    15  FILLER      PIC S9(3)V99   COMP-3     VALUE +2.75.
    15  FILLER      PIC S9(3)V99   COMP-3     VALUE +2.25.
10  FILLER.
```

```
            15  FILLER     PIC  S9(3)      COMP-3     VALUE  +100.
            15  FILLER     PIC  S9(3)V99   COMP-3     VALUE  +3.00.
            15  FILLER     PIC  S9(3)V99   COMP-3     VALUE  +4.23.
            15  FILLER     PIC  S9(3)V99   COMP-3     VALUE  +4.16.
        10  FILLER.
            15  FILLER     PIC  S9(3)      COMP-3     VALUE  +999.
            15  FILLER     PIC  S9(3)V99   COMP-3     VALUE  +3.50.
            15  FILLER     PIC  S9(3)V99   COMP-3     VALUE  +5.45.
            15  FILLER     PIC  S9(3)V99   COMP-3     VALUE  +5.20.
    05  FREIGHT-CHARGES-TABLE REDEFINES FREIGHT-CHARGE-VALUES.
        10  WEIGHT-RANGE              OCCURS 3 TIMES.
            15  WEIGHT-LIMIT          PIC  S9(3)  COMP-3.
            15  DESTINATION           OCCURS 3 TIMES.
                20  FREIGHT-CHARGE    PIC  S9(3)V99  COMP-3.
```

## التحميل بعبارة READ

هذا الإجراء – بمميزاته – متماثل مع الجدول ذى البعد الواحد ( انظر القسم ١٠ – ٣ ) .

مثال ١٠ – ٢١ :

يحمل جدول مثال (١٠ – ١٢) من ملف المدخلات :

```
FD  TABLE-VALUES-FILE...
01  TABLE-VALUES-RECORD.
    05  WEIGHT-VALUE          PIC  S9(3)          COMP-3.
    05  FREIGHT-VALUE         PIC  S9(3)V99       COMP-3.
                              OCCURS 3 TIMES.
```

أى إن كل سجل TABLE - VALUES - RECORD يوضع فى عنصر WEIGHT - RANGE ، بحيث يوضع WEIGHT - VALUE فى WEIGHT - LIMIT ، وتوضع القيم لثلاث من FREIGHT - VALUE فى المواقع الثلاث FREIGHT - CHARGES .

وتؤدى الشفرة التالية العمل .

```
 . . . . . . . . . . . . . . . . . . . . . . . . .
 OPEN  INPUT  TABLE-VALUES-FILE
 MOVE "NO" TO END-FILE-SW
 MOVE 1 TO NUMBER-WEIGHT-ENTRIES
 PERFORM GET-TABLE-RECORD
 PERFORM LOAD-TABLE
     UNTIL END-FILE-SW EQUAL "YES" OR NUMBER-WEIGHT-ENTRIES
         GREATER THAN SIZE-OF-WEIGHT-TABLE
 IF END-FILE-SW EQUAL "NO"
     DISPLAY "FREIGHT-CHARGES-TABLE TOO SMALL: CHANGE PROGRAM"
         UPON OPERATOR-MESSAGE-DEVICE
 .
 SUBTRACT 1 FROM NUMBER-WEIGHT-ENTRIES
 CLOSE TABLE-VALUES-FILE
 . . . . . . . . . . . . . . . . . . . . . . . . .
```

```
GET-TABLE-RECORD.
    READ TABLE-VALUES-FILE
        AT END
            MOVE "YES" TO END-FILE-SW

LOAD-TABLE.
    MOVE WEIGHT-VALUE TO WEIGHT-LIMIT (NUMBER-WEIGHT-ENTRIES)
    PERFORM MOVE-FREIGHT-CHARGES
        VARYING NUMBER-FREIGHT-CHARGES FROM 1 BY 1
            UNTIL NUMBER-FREIGHT-CHARGES GREATER THAN
                SIZE-OF-FREIGHT-TABLE
    PERFORM GET-TABLE-RECORD
    ADD 1 TO NUMBER-WEIGHT-ENTRIES

MOVE-FREIGHT-CHARGES.
    MOVE FREIGHT-VALUE (NUMBER-FREIGHT-CHARGES) TO
        FREIGHT-CHARGE (NUMBER-WEIGHT-ENTRIES NUMBER-FREIGHT-CHARGES)
```

والمعرف الدليلان وحدودهما فى قسم مخزن العمل :

```
01  TABLE-DIMENSIONS.
    05  NUMBER-WEIGHT-ENTRIES     PIC S9(5)  COMP SYNC.
    05  NUMBER-FREIGHT-CHARGES    PIC S9(5)  COMP SYNC.
    05  SIZE-OF-WEIGHT-TABLE      PIC S9(5)  COMP SYNC   VALUE +3.
    05  SIZE-OF-FREIGHT-TABLE     PIC S9(5)  COMP SYNC   VALUE +3.
```

لاحظ أنه بذل كل جهد لجعل البرنامج مرنا وسهلاً فى تغييره بقدر الإمكان . وعلى هذا .. تم حساب عدد عناصر الأوزان النشطة (كعدد سجلات منطقية فى حلف مدخلات ) ، بحيث يمكن ان يضبط البرنامج نفسه بطريقة مناسبة بين 1 و 3 لمحتويات WEIGHT - RANGE . والأكثر هو ظهور SIZE - OF - WEIGHT - TABLE و SIZE - OF - FREIGHT - TABLE فى الشفرة كعناصر بيانات ، بحيثإانهإاذا كان حجم الجدول مطلوب تغييره.. فلن يحدث تعديلاً إلا فى قيم VALUE , OC-CURS المناسبه فقط . (انظر المسألة ١٠ – ٧٨) .

## ١٠ – ٦ جداول متغيرة الطول :

## جزء الحدوث طبقاً لـ . . .

حتى الآن تعرضنا لجداول لها عدد ثابت من العناصر ، ويخصص لها عادة مكان يكفى أقصى عدد متوقع لعناصرها ، مع استخدام عداد counter لعد العدد الفعلى للعناصر النشطه active ، ولا توجد فى الكوبل طريقة لحفظ الذاكرة الرئيسية ، عندما يكون الجدول مملوءا بالكامل . ولكن عندما يتم إخراج الجداول على قرص أو شريط .. يقدم الكوبل آلية تجعل العناصر النشطة فقط جزءا من السجل المنطقى ، محافظة بذلك على مساحة القرص أو الشريط .

**مثال ١٠ – ٢٢ :**

اعتبر الملف التالى :

```
FD  PRODUCT-MASTER-FILE...
01  PRODUCT-MASTER-RECORD.
    05  PRODUCT-ID                PIC X(5).
    05  PRODUCT-DESCRIPTION       PIC X(20).
    05  NUMBER-OF-SUPPLIERS       PIC S9(2)  COMP.
    05  SUPPLIER-INFORMATION      OCCURS 1 TO 50 TIMES
                                  DEPENDING ON NUMBER-OF-SUPPLIERS.
        10  SUPPLIER-ID           PIC X(4).
        10  SUPPLIER-NAME         PIC X(20).
        10  SUPPLIER-PRICE        PIC S9(3)V99  COMP-3.
        10  SUPPLIER-TERMS        PIC X(2).
```

يحدد جزء OCCURS ... DEPENDING للمترجم أنه بالرغم من أن أقصى حجم للجدول هو 50 عنصراً ، إلا أن الحجم النشط active للجدول (الحجم الفعال على القرص أو الشريط) يتحدد عن طريق محتويات NUMBER - OF - SUPPLIERS ، والذى يمكن أن يتراوح من واحد إلى خمسين .

عندما يكون **جدول متغير الطول** variable - length table (كما هو معرف بجزء OCCURS ... DEPENDING .. ) جزءا من سجل منطقى ، يحتوى الملف على سجلات متغيرة الطول (الفصل الخامس) .

تكوين .... OCCURS .. DEPENDENG لجداول متغيرة الطول ، هو ما يلى :

<u>OCCURS</u> integer-1 <u>TO</u> integer-2 TIMES
<u>DEPENDING</u> ON data-name

حيث integer - 1 يجب أن يكون 1 على الأقل ، و أقل من integer - 2 . data - name هو عنصر عدديًا (له أى شكل من أشكال USAGE ) والذى يمكن - أثناء التنفيذ - أن يفترض قيما صحيحة تقع بين integer - 1 , integer - 2 بما فى ذلك الحدود . والقيمة الحالية لاسم البيانات data - name هى التى تحدد عدد عناصر الجدول النشطة حاليا (وبالتالى تحدد المكان اللازم لسجل منطقى متغير الطول فى التخزين المساعد ) . لاحظ أنه يجب أن يكون فى الجدول متغير الطول عنصر واحد على الاقل (انظر ملحق حـ بالنسبه لكوبل 1985 النمطى) .

عندما يوجد جدول متغير الطول فى سجل ... فالمطلوب أن يكون هو آخر عنصر فى السجل ؛ فإذا كانت الجداول متداخلة .. فإن الجدول متغير الطول هو جدولاًاعلى مستوى فقط . وفى كلمات أخرى .. لا يمكن أن يكون جزء OCCURS .. DE-PENDING تابعا لأى جزء OCCURS آخر .

## نقل جداول متغيرة الطول

إذا كان من المرغوب فيه نقل جدول متغير الطول من منطقة بياناتإالى منطقة أخرى.،، فيجب أن توضع قيم عنصرى DE-PENDING متساوية ، قبل تنفيذ MOVE . (انظر ملحق حـ بالنسبة الى تعديلات كوبل 1985 النمطى) .

**مثال ١٠ - ٢٣ :**

تحتوى الطريقة الصحيحة لوضع سجل منطقى - فى مخزن العمل - على جدول متغير الطول :

```
FD   ASSEMBLY-FILE...
01   ASSEMBLY-RECORD.
     05  ITEM-ID                         PIC X(6).
     05  NUMBER-SUBASSEMBLIES            PIC S9(2)      COMP.
     05  SUBASSEMBLY-INFO                OCCURS 1 TO 25 TIMES
                                         DEPENDING ON
                                             NUMBER-SUBASSEMBLIES.
         10  SUBPART-ID                  PIC X(6).
         10  WAREHOUSE-LOCATION          PIC XX.
         10  QUANTITY-AVAILABLE          PIC S9(5)      COMP-3.
. . . . . . . . . . . . . . . . . . . . . . . . . . . .
WORKING-STORAGE SECTION.
01  WS-ASSEMBLY-RECORD.
     05  WS-ITEM-ID                      PIC X(6).
     05  WS-NUMBER-SUBASSEMBLIES         PIC S9(2)      COMP.
     05  WS-SUBASSEMBLY-INFO             OCCURS 1 TO 25 TIMES
                                         DEPENDING ON
                                             WS-NUMBER-SUBASSEMBLIES.
         10  WS-SUBPART-ID               PIC X(6).
         10  WS-WAREHOUSE-LOCATION       PIC XX.
         10  WS-QUANTITY-AVAILABLE       PIC S9(5)      COMP-3.
. . . . . . . . . . . . . . . . . . . . . . . . . . . .
     READ ASSEMBLY-FILE  AT END...
     MOVE NUMBER-SUBASSEMBLIES           TO WS-NUMBER-SUBASSEMBLIES
     MOVE ASSEMBLY-RECORD                TO WS-ASSEMBLY-RECORD
```

لاحظ أن عبارة READ ASSEMBLY - FILE INTO WS - ASSEMBLY - RECORD بسيطة لا تؤدى العمل ؛ حيث إنها تكافىء :

```
READ ASSEMBLY-FILE...
MOVE ASSEMBLY-RECORD TO WS-ASSEMBLY-RECORD
```

والتى لا تساوى WS - NUMBER - SUBASSEMBLIES ، مـع NUMBER - SUBASSEMBLIES ، وذلك قبل تنفيذ عبارة النقل . ويمكن أن تكون نتيجة النقل حذفاً أو إضافة فى WS - ASSEMBLY - RECORD طبقا للمحتويات السابقة فى WS - NUMBER - SUBASSEMBLIES .

## توسيع أو تضييق الجداول

يمكن إضافة أو حذف عناصر نشطة إلى ومن جدول متغير الطول ؛ بشرط عدم تعدى حدود TIMES فى جزء OCCURS. فإذا كان مطلوبا إضافة أحد العناصر .. فيجب أن تزداد قيمة عنصر DEPENDING بمقدار 1 .

يجب ألا تجدى التغييرات فى جدول متغير الطول التى تؤثر على حجم الجدول فى الذاكرة الاحتياطية لإدخال السجل المنطقى (حيث إنها يمكن أن تثير أى سجلات منطقية تالية فى الملف المستخدم للتجميع) . ويمكن إجراء مثل هذه التغييرات فى نسخة مخزن العمل لسجل المدخلات ، أو فى الذاكرة الاحتياطية لسجل المخرجات .

**مثال ١٠ - ٢٤ :**

لإضافة معلومات من :

```
FD  TRANSACTION-FILE...
01  TRANSACTION-RECORD.
    05  INPUT-ITEM-ID                  PIC X(6).
    05  INPUT-SUBPART-ID               PIC X(6).
    05  INPUT-WAREHOUSE-LOCATION       PIC XX.
    05  INPUT-QUANTITY-AVAILABLE       PIC S9(5).
```

إلى WS - ASSEMBLY - RECORD فى مثال (١٠ - ٢٣) .. فإننا نكتب الشفرة التالية :

```
ADD 1 TO WS-NUMBER-SUBASSEMBLIES
MOVE INPUT-SUBPART-ID TO
    WS-SUBPART-ID (WS-NUMBER-SUBASSEMBLIES)
MOVE INPUT-WAREHOUSE-LOCATION TO
    WS-WAREHOUSE-LOCATION (WS-NUMBER-SUBASSEMBLIES)
MOVE INPUT-QUANTITY-AVAILABLE TO
    WS-QUANTITY-AVAILABLE (WS-NUMBER-SUBASSEMBLIES)
```

لاحظ أن عنصر DEPENDING ازداد بمقدار 1 أولا ؛ مضافاً إلى ذلك أن الجدول يتسع للعنصر الجديد . وعلى ذلك يخدم عنصر DEPENDING كدليل لتوقيع العنصر الجديد فى نهاية الجدول (أسهل موقع لوضعه انظر مثال (١٠ - ٣٧) لمعرفة طريقة أخرى) .

**مثال ١٠ - ٢٥ :**

افرض أن INPUT - SUBPART - ID يعرف جزءاً مجمعاً تجميعاً جزئيا ، مطلوب حذفه من WS - ASSEMBLY - RECORD فى مثال (١٠ - ٢٣) .

(أ) فى الحذف المنطقى logical deletion .. ننقل فراغات (أو أى رمز حذف deletion code آخر ) الى عنصر الجدول الذى يتفق مع INPUT - SUBPART - ID ، وتحدد الفراغات أن العنصر غير نشيط لاى برنامج ، يجرى تشغيلا على الجدول. إلا أننا لم نقلل WS - NUMBER - SUBASSEMBLIES ؛ حيث إن العنصر لم يحذف واقعيا physically .

```
MOVE "YES" TO ENTRY-IN-TABLE
MOVE 1 TO TABLE-SUBSCRIPT
PERFORM LOCATE-SUBPART
    UNTIL ENTRY-IN-TABLE EQUAL "NO"
    OR INPUT-SUBPART-ID EQUAL
        WS-SUBPART-ID (TABLE-SUBSCRIPT)
```

```
    IF ENTRY-IN-TABLE EQUAL "NO"
        PERFORM ERROR-ROUTINE
    ELSE
        MOVE SPACES TO WS-SUBPART-ID (TABLE-SUBSCRIPT)
    .
. . . . . . . . . . . . . . . . . . . . . . . .
LOCATE-SUBPART.
    IF TABLE-SUBSCRIPT EQUAL WS-NUMBER-SUBASSEMBLIES
        MOVE "NO" TO ENTRY-IN-TABLE
    ELSE
        ADD 1 TO TABLE-SUBSCRIPT
    .
```

بوجود فراغات فى WS - SUBPART - ID (j) .. فان : أى برنامج مستقبلى يجرى تشغيلا على الجدول ، يهمل كذلك WS- QUANTITY - AVAILABLE (j) و WS - WAREHOUSE - LOCATION (j) .

(ب) فى الحذف الواقعى physical deletion .. تنقل كل العناصر التى تتبع العنصر المتفق مع INPUT - SUBPART - ID موقعا واحدا (فى اتجاه بداية الجدول) ، وتقل قيمة عنصر DEPENDING بمقدار واحد ، حيث حذف عنصر واحد حذفاً واقعياً .

```
    MOVE "YES" TO ENTRY-IN-TABLE
    MOVE 1 TO TABLE-SUBSCRIPT
    PERFORM LOCATE-SUBPART
        UNTIL ENTRY-IN-TABLE EQUAL "NO"
            OR INPUT-SUBPART-ID EQUAL
                WS-SUBPART-ID (TABLE-SUBSCRIPT)
    IF ENTRY-IN-TABLE EQUAL "NO"
        PERFORM ERROR-ROUTINE
    ELSE
        PERFORM MOVE-UP-TABLE-ENTRIES
            VARYING MOVING-SUBSCRIPT FROM TABLE-SUBSCRIPT BY 1
                UNTIL MOVING-SUBSCRIPT EQUAL WS-NUMBER-SUBASSEMBLIES
        SUBTRACT 1 FROM WS-NUMBER-SUBASSEMBLIES
    .
. . . . . . . . . . . . . . . . . . . . . . . .
LOCATE-SUBPART.
    IF TABLE-SUBSCRIPT EQUAL WS-NUMBER-SUBASSEMBLIES
        MOVE "NO" TO ENTRY-IN-TABLE
    ELSE
        ADD 1 TO TABLE-SUBSCRIPT
    .
MOVE-UP-TABLE-ENTRIES.
    ADD 1 MOVING-SUBSCRIPT GIVING MOVING-SUBSCRIPT-PLUS-1
    MOVE WS-SUBASSEMBLY-INFO (MOVING-SUBSCRIPT-PLUS-1) TO
        WS-SUBASSEMBLY-INFO (MOVING-SUBSCRIPT)
    .
```

لاحظ كيف وضعت PERFORM LOCATE - SUBPART UNTIL الدليل TABLE - SUBSCRIPT فى رقم الموقع الخاص بالعنصر الذى سيحذف إذا كان العنصر موجودا فى الجدول) .يجب أن تنقل كل العناصر التى تقع بين TABLE - SUBSCRIPT ونهاية الجدول (فى WS - NUMBER - SUBASSEMBLIES ) موقعا واحدا ، لضغط مكان العنصر المحذوف من الجدول . ويحدث ذلك فى MOVE - UP - TABLE - ENTIRIES ؛ حيث ينقل العنصر عند MOVING - SUBSCRIPT +1 الى العنصر عند MOVING - SUBSCRIPT . يتغير MOVING - SUBSCRIPT من TABLE - SUBSCRIPT الى WS - NUMBER - SUBASSEMBLIES - 1 (تذكر أنه عند استخدام EQUAL بدلا من GREATER فى عبارة PERFORM ... VARYING ... UNTIL .. فإن آخر قيمة لعنصر VARYING ، والتى ينفذ لها المقطع فعلا تقل "1" عن القيمة المحددة للتوقف) .و هذا ضرورى بسبب أن مواقع العناصر التى تنقل تحددت بواسطة MOVING - SUBSCRIPT مضافا «١» إليها . وحيث إن الدليل لا يمكن أن يكون تعبيرا .. فمن الضرورى تعريف عنصر بيانات إضافى MOVING - SUBSCRIPT - PLUS - 1 ، يحسب فيه MOVING - SUBSCRIPT + 1 . بعد نقل كل العناصر المناسبة موقعا واحدا .. تقل قيمة عنصر DEPENDING بمقدار «1» لتحديد أن عناصر الجدول قد قلت الآن . يفضل الحذف الواقعى عن الحذف المنطقى للجداول متغيرة الطول ؛ لأنه يقلل حجم مواقع التخزين المساعد اللازمة لحفظ الجدول .

## ١٠ - ٧ الفهارس

عندما تستخدم الأدلة فى تعريف عناصر الجدول .. يجب أن ينتج المترجم تعليمات لغة آلة لتحويل قيمة الدليل الى العنوان الفعلى للذاكرة وذلك للعنصر المطلوب . الفهارس indexes هى أسماء بيانات كوبل ، يمكن استخدامها بدلا من الدلائل فى جعل حسابات عناوين لغة الآلة أكثر كفاءة (انظر المسألتين ٧٤ و ٧٥ من هذا الفصل) . وقيمة الفهرس هى إزاحة displacement فى الجدول ؛ أى إنها رقم عندما يضاف الى عنوان بداية الجدول .. ينتج عنه عنوان العنصر المطلوب . الفهارس أكثر كفاءة من الدلائل ، ويجب استخدامها كلما أمكن .

تعرف الفهارس المستخدمة بدلا من الدلائل لأحد الجداول فى جزء OCCURS الذى يعرف الجدول ، ولا تحتاج أى تعريف خاص بها ( على عكس الدلائل ) ، ويبين تكوين جزء OCCURS عند تعريف فهارس فى شكل (١٠ - ٢) .

```
OCCURS integer-1 [TO integer-2] TIMES
    [DEPENDING ON data-name-1]
    [INDEXED BY index-name-1 [index-name-2] . . .]
```

شكل ( ١٠ - ٢ )

أصبحت index - name - 1 , index - name - 2 و ... إلخ ، متاحة لاستخدامها بدلاً من الدلائل عند الاتصال بمحتويات الجدول .

مثال ١٠ - ٢٦ :

```
01  SALES-FIGURES-BY-STATE.
    05  STATE-SALES          OCCURS 50 TIMES
                             INDEXED BY STATE-NUMBER.
        10  STATE-ID         PIC X(2).
        10  SALES-AMOUNT     PIC S9(5)V99   COMP-3.
```

هنا تم تعريف جدول مثال (١٠ - ٤) باستخدام الفهارس . يتاح الفهرس STATE - NUMBER الآن لاستخدامه بدلا من STATE - SALES - SUB . لاحظ أن الفهرس يصاحبه جدول خاص particular ، ويجب ألا يستخدم STATE - NUMBER فى الاتصال بأى جدول آخر، إلا إذا كان لهذا الجدول نفس الهيكل تماما ( USAGE , PICTURE ، والمستويات وعدد المحتويات ، وغيرها ) مثل STATE - SALES . لاحظ كذلك أنه لا توجد حاجة لتعريف أكثر للفهرس (غير مسموح بذلك أيضا)

لا يمكن معاملة الفهارس مثل الدلائل ؛ إذ يمكن وضع قيمة ابتدائية للفهرس ، أو تغيير الفهرس (١) باستخدام عبارة SET فقط (٢) ، أو عبارة PERFORM (٣) ، أو عبارة SEARCH . (انظر القسم ١٠ - ٨) .

يمكن استخدام عبارة SET فى وضع قيمة ابتدائية لفهرس ، سواء كانت قيمة فهرس آخر ، أو عنصر بيانات عدديًا ، أو ثابتاً عدديًا . وعكسيا .. يمكن استخدام عبارة SET فى وضع عنصر بيانات عددى فى القيمة الحالية لأحد الفهارس ، وتكوينها هو كما يلى :

$$\underline{\text{SET}} \begin{Bmatrix} \text{index-name-1} \\ \text{data-name-1} \end{Bmatrix} \begin{bmatrix} \text{index-name-2} \\ \text{data-name-2} \end{bmatrix} \cdots \underline{\text{TO}} \begin{Bmatrix} \text{index-name-3} \\ \text{identifier-3} \\ \text{literal} \end{Bmatrix}$$

مثال ١٠ - ٢٧ :

- SET STATE-NUMBER TO 3

تحول عبارة SET القيمة 3 الى إزاحة العنصر الثالث فى STATE - SALES ( انظر مثال ١٠ - ٢٦ ) .

- SET STATE-NUMBER TO SOME-OTHER-INDEX

إذا أ شار SOME - OTHER - INDEX إلى العنصر K فى SOME - OTHER - TABLE .. فإن STATE - NUM-BER يشير الى العنصر K فى STATE - SALES .

- SET STATE-NUMBER TO A-SUBSCRIPT

أعد STATE - NUMBER ليشير الى عنصر الجدول ، الذى يكون رقم موضعه فى A - SUBSCRIPT حاليا . ويمكن أن يعرف A - SUBSCRIPT كما يلى :

```
05  A-SUBSCRIPT    PIC S9(5)    COMP SYNC.
```

- SET A-SUBSCRIPT TO STATE-NUMBER

ينقل رقم الموقع المناظر للإزاحة المخزنة حاليا فى STATE - NUMBER إلى A - SUBSCRIPT .

يجب استخدام عبارة SET كذلك فى زيادة قيمة الفهرس أو تعديلها :

$$\underline{\text{SET}}\ \text{index-name-1 [index-name-2]}\ldots \left\{\begin{matrix}\underline{\text{UP BY}}\\ \underline{\text{DOWN BY}}\end{matrix}\right\} \left\{\begin{matrix}\text{data-name}\\ \text{literal}\end{matrix}\right\}$$

(MOVE . . . , ADD . . . , or SUBTRACT . . . *may not be used* with indexes.)

مثال ١٠ – ٢٨ :

- SET STATE-NUMBER DOWN BY 1

تقلل الإزاحة فى STATE - NUMBER بمقدار طول عنصر الجدول ، بحيث إن STATE - NUMPER يشير إلى العنصر السابق .

- SET STATE-NUMBER UP BY A-SUBSCRIPT

تزداد قيمة STATE - NUMBER بمقدار الإزاحة التى تجعله يشير إلى أن عدد العناصر يساوى المحتوى الحالى - A SUBSCRIPT (مثل إذا بدأ STATE - NUMBER فى الإشاره إلى ثالث عنصر ، واحتوى A - SUBSCRIPT على القيمة 5 .. فعند ذلك يشير STATE - NUMBER إلى العنصر الثامن ) .

يمكن استخدام الفهارس فى عبارات PERFORM .. وفى شروط IF كعناصر بيانات عددية صحيحة تماما .

**مثال ١٠ – ٢٩ :**

أوجد إجمالى قيمة المبيعات فى مثال (١٠ – ٢٦) .

```
01  SALES-FIGURES-BY-STATE.
    05  STATE-SALES           OCCURS 50 TIMES INDEXED BY STATE-NUMBER.
        10  STATE-ID          PIC X(2).
        10  SALES-AMOUNT      PIC S9(5)V99     COMP-3.
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    MOVE ZERO TO TOTAL-SALES
    PERFORM ADD-UP-STATE-SALES
        VARYING STATE-NUMBER FROM 1 BY 1
        UNTIL STATE-NUMBER GREATER THAN 50
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
ADD-UP-STATE-SALES.
    ADD SALES-AMOUNT (STATE-NUMBER) TO TOTAL-SALES
```

لاحظ أن STATE - NUMBER يستخدم مثل الدليل تماماً. وفى الواقع .. بالنظر فقط إلى جزء OCCURS ، يمكن القول بانه ليس دليلا . وبالرغم من ذلك .. تنتج عن الفهرس STATE - NUMBER لغة آلة أكثر كفاءة .

**مثال ١٠ – ٣٠ :**

أعد كتابة LOOK- UP - STATUS من مثال (١٠ – ١٠) : مستخدما فهرسا بدلا من الدليل .

```
LOOK-UP-STATUS.
    SET STATUS-TABLE-INDEX TO 1
    MOVE "YES" TO VALID-STATUS-SW
    PERFORM CHECK-ENTRY
        UNTIL VALID-STATUS-SW EQUAL "NO"
            OR STATUS-CODE (STATUS-TABLE-INDEX) EQUAL INPUT-STATUS-CODE
CHECK-ENTRY.
    IF STATUS-TABLE-INDEX EQUAL 40
        MOVE "NO" TO VALID-STATUS-SW
    ELSE
        SET STATUS-TABLE-INDEX UP BY 1
```

يجب وضع SET الفهرس ، وإلا فإنه يصبح دليلا .

مثال ١٠ – ٣١ :

اعد كتابة مثال ( ١٠ – ١٨ ) : مستخدما فهرساً بدلا من الدليل .

```
FD  SALES-INPUT-FILE...
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
01  TOTAL-SALES-AMOUNTS-TABLE.
    05  REGION                      OCCURS 7 TIMES
                                    INDEXED BY REGION-INDEX.
        10  DEPARTMENT              OCCURS 12 TIMES
                                    INDEXED BY DEPARTMENT-INDEX.
            15  TOTAL-SALES-AMOUNT  PIC S9(5)V99  COMP.
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
ELSE
    SET REGION-INDEX TO REGION-ID
    SET DEPARTMENT-INDEX TO DEPARTMENT-ID
    ADD SALES-AMOUNT TO TOTAL-SALES (REGION-INDEX DEPARTMENT-INDEX)

PERFORM GET-SALES-RECORD
```

## الفهرسة النسبية

يمكن جمع أو طرح ثابت عددى ( وثوابت عددية فقط ) من الفهرس للإشارة إلى عنصر بالنسبة إلى القيمة الحالية للفهرس . وعادة ما يكون هذا مقنعا ، عند نقل عناصر الجدول هنا وهناك داخل الجدول .

مثال ١٠ – ٣٢ :

أفرض أنه يرغب فى نقل عنصر من الموقع K إلى الموقع K-5 ، باستخدام الأدلة يمكننا أن نعرف.

```
01  TABLE-SUBSCRIPTS.
    05  K              PIC S9(5)  COMP SYNC.
    05  K-MINUS-FIVE   PIC S9(5)  COMP SYNC.
    . . . . . . . . . . . . . . . . . . . . . . . . . .
    SUBTRACT 5 FROM K GIVING K-MINUS-FIVE
    MOVE TABLE-ENTRY (K) TO TABLE-ENTRY (K-MINUS-FIVE)
```

ونستطيع بالفهرسة استخدام جزءا أكثر سهولة :

```
SET TABLE-INDEX TO K
MOVE TABLE-ENTRY (TABLE-INDEX) TO TABLE-ENTRY (TABLE-INDEX - 5)
```

## ١٠ - ٨ البحث المتتالى فى الجداول وفعل «ابحث»

سبق أن أوضحت الأقسام السابقة كيفية برمجة البحث فى جدول باستخدام دورة PERFORM . الا أن كـوبـل ANS النمطى يقوم فعل بحث SEARCH خاص ، أسهل فى استخدامه وأكثر كفاءة كذلك ؛ فإذا احتوى الجدول على n عنصر ..فإن البحث المتتالى (الخطى) يفحص العناصر بالترتيب K ثم K+1 ثم K+2 ... ثم n-1 ثم n . (عادة ما تكون مساوية 1 (لعبارة البحث المتتالى التكوين الموجود فى شكل (١٠ - ٣) ؛ يجب أن يكون identifier هو اسم الجدول المراد البحث فيه (أى إنه عنصر بيانات موصوف باستخدام OCCURS ) . يحدد جزء AT END المطلوب عمله ، عندما يفشل البحث (أى عندما لايتحقق أى شرط من شروط WHEN أثناء البحث) .

```
SEARCH identifier-1 [VARYING {identifier-2  }]
                             {index-name-1  }
    [AT END imperative-statement(s)-1]
     WHEN condition-1 {imperative-statement(s)-2}
                      {NEXT SENTENCE            }
    [WHEN condition-2 {imperative-statement(s)-3}] ...
                      {NEXT SENTENCE            }
```

شكل ( ١٠ - ٣ )

مثال ١٠ - ٣٢ :

أعد مثال (١٠ - ١٠) مستخدما أمر البحث بدلا من أمر التنفيذ .

```
01  VALID-STATUS-CODES-TABLE.
    05  STATUS-CODE                  PIC X(3)
                                     OCCURS 40 TIMES
                                     INDEXED BY VALID-CODE-INDEX.
. . . . . . . . . . . . . . . . . . . . . . . . . . .
LOOK-UP-STATUS.
    SET VALID-CODE-INDEX TO 1
    SEARCH STATUS-CODE
        AT END
            MOVE "NO" TO INPUT-CODE-OK-SW
        WHEN INPUT-STATUS-CODE EQUAL STATUS-CODE (VALID-CODE-INDEX)
            MOVE "YES" TO INPUT-CODE-OK-SW
```

لاحظ أنه يجب تعريف الجدول بفهرس index ، إذا ما كان فعل SEARCH مستخدما . كما يجب أن يضع المبرمج فهرس البحث عند العنصر الذى يراد للبحث أن يبدأ منه (عادة ما يكون «1» ) . وتعمل عبارة SEARCH السابقة كما يلى :

(١) نظراً لأن جزء VARYING غير محدد .. تستخدم SEARCH أول فهرس معرف لجدول STATUS - CODE . وفى هذه الحالة يكون أول فهرس (والأول فقط) VALID - CODE - INDEX .

(٢) تقارن قيمة فهرس SEARCH مع عدد عناصر الجدول ( هنا 40 ) . إذا زادت قيمة الفهرس عن حجم الجدول .. ينفذ عبارة AT END ، ويستمر – عند ذلك – التنفيذ بأول عبارة بعد النقطة التى تنهى عبارة SEARCH .

(٣) إذا لم يتعد الفهرس حجم الجدول .. يتم تقويم شروط WHEN (بالترتيب الذى كتبت به) . ويتسبب أول شرط WHEN متحقق فى تنفيذ العبارة المصاحبة له . وعندها .. ينتهى البحث (ويستمر التنفيذ بأول عبارة تتبع النقطة التى تنهى عبارة البحث) .

(٤) إذا لم يتحقق أى شرط من شروط WHEN .. يزداد فهرس SEARCH بمقدار عنصر واحد ، ويعود تنفيذ عبارة البحث إلى الخطوة رقم (٢) المذكورة أعلاه .

**مثال ١٠ – ٣٤ :**

باستخدام SEARCH ... VARYING .. نستطيع فحص جداول متوازية parallel tables ، وهى جداول تحتوى عناصرها المتناظرة على بيانات عن نفس العنصر . افرض أن لدينا الجداول المتوازية التالية :

```
01  DEDUCTION-CODES-TABLE.
    05  DEDUCT-CODE              PIC X
                                 OCCURS 30 TIMES
                                 INDEXED BY CODE-INDEX.

01  DEDUCTIONS-INFO-TABLE.
    05  PAYROLL-DEDUCTIONS       OCCURS 30 TIMES
                                 INDEXED BY INFO-INDEX.
        10  DEDUCT-DESCRIPTION   PIC X(25).
        10  DEDUCT-AMOUNT        PIC S99V99     COMP-3.
```

حيث (k) DEDUCT - CODE - يناظر (k) PAYROLL - DEDUCTIONS . بمعرفة أحد المدخلات - PAYROLL CODE .. فإننا نقوم بتشغيل خصومات الراتب باستخدام SEARCH ... VARYING :

```
SET CODE-INDEX TO 1
SET INFO-INDEX TO 1
SEARCH DEDUCT-CODE
    VARYING INFO-INDEX
    AT END
        MOVE "YES" TO PAYROLL-ERROR-SW
    WHEN DEDUCT-CODE (CODE-INDEX) EQUAL PAYROLL-CODE
        MOVE DEDUCT-DESCRIPTION (INFO-INDEX) TO OUTPUT-AREA
        SUBTRACT DEDUCT-AMOUNT (INFO-INDEX) FROM NET-PAY
```

حيث إن الجدولين لهما هيكلان مختلفان .. فمن الضروري استخدام فهرسين . وعندما يحدد جزء VARYING فهرسا ينتمي إلى جدول غير جدول SEARCH (١) تزداد قيمة أول فهرس محدد في جدول SEARCH (وهو CODE - INDEX) اثناء البحث . (٢) كلما ازدادت قيمة أول فهرس لجدول SEARCH... إزدادت كذلك قيمة فهرس VARYING ( هو INFO - INDEX ) . وعلى هذا فإن فهرس VARYING يتبع فهرس SEARCH ؛ بحيث يشيران دائما إلى المحتويات المتناظرة مع بعضها البعض ، بافتراض أنها وضعت لـ 'SET نفس القيمة قبل تنفيذ عبارة البحث .

إذا كان للجدول أكثر من فهرس واحد معرف له ، وتسمى VARYING فهرسا غير الفهرس الأول ؛ فيستخدم هذا الفهرس في البحث .

حيث إن SEARCH ترتبط مع جزء OCCURS واحد في نفس الوقت .. فلابد من كتابة شفرة خاصة لإجراء بحث كامل في جدول ذى بعدين . وعادة ما يكفي وضع عبارة بحث في مقطع PERFORM ، مع ترك عبارة PERFORM .. تغير VARY أول فهرس (ويتغير الثاني بواسطة عبارة البحث) .

**مثال ١٠ – ٣٥ :**

لتحديد INPUT - COURSE PIC X (5) في الجدول :

```
01  STUDENT-TRANSCRIPT.
    05  STUDENT-ID                PIC X(9).
    05  SEMESTERS-ATTENDED        PIC S9       COMP.
    05  SEMESTER-INFORMATION      OCCURS 1 TO 8 TIMES
                                  DEPENDING ON SEMESTERS-ATTENDED
                                  INDEXED BY SEMESTER-INDEX.
        10  SEMESTER-DATE         PIC X(6).
        10  NUMBER-COURSES        PIC S9       COMP.
        10  COURSE-INFO           OCCURS 6 TIMES
                                  INDEXED BY COURSE-INDEX.
            15  COURSE-ID         PIC X(5).
            15  CREDITS           PIC S9V9     COMP-3.
            15  GRADE             PIC S9       COMP-3.
```

ونكتب الشفرة على النحو التالي :

```
. . . . . . . . . . . . . . . . . . . . . . . . . . . .
MOVE "NO" TO FOUND-IT-SW
PERFORM LOCATE-COURSE
    VARYING SEMESTER-INDEX FROM 1 BY 1
        UNTIL SEMESTER-INDEX GREATER THAN SEMESTERS-ATTENDED
            OR FOUND-IT-SW EQUAL "YES"
IF FOUND-IT-SW EQUAL "YES"
    SET SEMESTER-INDEX DOWN BY 1
    PERFORM PROCESS-COURSE-INFO
```

```
    ELSE
        PERFORM COURSE-NOT-FOUND-ERROR

. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
LOCATE-COURSE.
    SET COURSE-INDEX TO 1
    SEARCH COURSE-INFO
        WHEN COURSE-INDEX GREATER THAN
             NUMBER-COURSES (SEMESTER-INDEX)
            NEXT SENTENCE
        WHEN COURSE-ID (SEMESTER-INDEX COURSE-INDEX)
            EQUAL INPUT-COURSE
            MOVE "YES" TO FOUND-IT-SW
```

## تعليقات :

(١) تنفذ عبارة البحث مرات ومرات مع تغيير SEMESTSR - INDEX من «1» الى SEMESTERS - ATTENDED ، وهو مداه الكامل من القيم الممكنة .

(٢) توضع SET القيمة «1» للفهرس COURSE - INDEX فى كل مرة ينفذ LOCATE - COURSE . وعلى هذا .. يبحث فى محتويات الجدول COURSE - INFO ، لكل قيمة من قيم SEMESTER - INDEX .

(٣) يفتقر اسم جدول البحث COURSE - INFO إلى وجود أدلة أو فهارس فى عبارة البحث ، وذلك بالرغم من أنه محتوى فى جدول SEMESTER - INFORMATION ، وعادة ما يكون له أدلة أو فهارس .

(٤) لا يعد عدم وجود جزء AT END مشاكله هنا ؛ فاذا تم الوصول إلى نهاية COURSE - INFO لقيمة معينة من قيم SEM-STER - INDEX .. فسوف يستمر التنفيذ بالعبارة التى تلى عبارة البحث ، وهى – فى هذه الحالة – نهاية المقطع . إلا أن الوصول إلى نهاية LOCATE - COURSE يتسبب فى زيادة عبارة PERFORM من SEMETER - INDEX ، وتنفذ المقطع مرة أخرى ؛ أى بحثا عن COURSE - INFO للفصل الدراسى التالى ، كما هو مطلوب .

(٥) يختبر جزء WHEN الأول النهاية المنطقية logical end لجدول COURSE - INFO ؛ فإذا كان هذا الشرط صحيحا .. تتسبب NEXT SENTENCE فى استمرار التنفيذ حتى نهاية المقطع ، مع زيادة قيمة SEMESTER - INDEX ، كما فى التعليق رقم (٤) .

(٦) إذا تحقق ثانى شرط WHEN .. توضع قيمة YES للمفتاح FOUND - IT - SW ؛ إلا أن PERFORM ... VARY-ING تزيد من SEMESTER - INDEX أولاً ، ثم تجرى تقويما لشرط UNTIL (FOUND - IT - SW EQUAL "YES") .

```
SET SEMESTER-INDEX DOWN BY 1
```

لا تجر الزيادة الإضافية .

## ١٠ - ٩ البحث الثنائى فى الجداول، وفعل ابحث الكل

خوارزمى البحث الثنائى binary search أكثر كفاءة من البحث التتابعى للجداول التى تكون كبيرة نسبيا ، والتى تكون مرتبة sorted ترتيبا تصاعديا ascending sequence ، أو ترتيبا تنازليا descending sequence ؛ طبقا لبعض حقول الجدول ، والذى يسمى حقلا رئيسيا ، أو مفتاح الترتيب key . وكلما ازداد حجم الجدول .. ازدادت أهمية البحث الثنائى عن البحث المتتابع . و يبدأ البحث الثنائى بفحص عنصر الوسط (يفسر بطريقة مناسبة إذا كان العدد زوجيا) . إذا كان العنصر المطلوب يقع فى منتصف الجدول .. فسوف ينتهى البحث ، إما إذا كان العنصر المطلوب أكبر من (له قيمة أكبر للمفتاح) عنصر الوسط ... فيجب أن يقع فى الجزء الأعلى من الجدول (بالنسبة للجداول المخزنة فى ترتيب تصاعدى) ، .. وبالمثل إذا كان العنصر المطلوب أقل من عنصر الوسط ... فيجب أن يقع النصف الأقل من الجدول . فى أى حالة من الحالتين .. يحذف نصف الجدول ؛ بدلاً من اخذه فى الاعتبار .

يستمر البحث بفحص عنصر الوسط لنصف الجدول المتبقى وتطبق نفس العملية ، مع التخلص من نصف الجدول الحالى . أخيرا .. لا يوجد إلا عنصراً واحد اً ، سواء كان هو العنصر المطلوب ، أو لم يكن العنصر المطلوب موجودا فى الجدول .

يبرمج البحث الثنائى فى الكوبل باستخدام عبارة ابحث الكل SEARCH ALL (شكل ١٠ - ٤) . يجب أن يكون identifi-er - 1 هو اسم الجدول (معرفا بجزء OCCURS) المراد البحث فيه ، كما يجب ألا يكون له دليل أو فهرس فى جزء SEARCH - ALL نفسه . يعمل جزء AT END كما يعمل فى البحث المتتابع ، بتنفيذه إذا فشل البحث الثنائى فى الجدول المحدد فقط . وإذا تحقق شرط WHEN أثناء البحث .. فتنفذ العبارة (أو العبارات) المصاحبة له وينتهى البحث. لاحظ أن شرط WHEN يجب أن يكون بنفس الصورة الموجودة فى شكل (١٠ - ٤) .

```
SEARCH ALL identifier-1
       [AT END imperative-statement(s)-1]
                                                    { identifier-2          }
    WHEN ascending/descending-key-item EQUAL { literal                }
                                                    { arithmetic-expression }
              { imperative-statement(s)-2 }
              { NEXT SENTENCE              }.
```

شكل ( ١٠ - ٤ )

بعد تنفيذ عبارة (عبارات) AT END ، أو عبارة (عبارات) WHEN (أو بعد تحقق شرط AT END دون أن يحدد شرط AT END فى العبارة) ، يستمر التنفيذ بالعبارة التى تلى النقطة .. التى تنهى عبارة SEARCH ALL مباشرة .

عندما يجرى تشغيل على جدول باستخدام عبارة SEARCH ALL .. يجب أن يحتوى جزء OCCURS الخاص به على جزء ASCENDING KEY ، أو DESCENDING KEY (شكل ١٠ - ٥) . من المهم ان يفهم إن هذا الجزء لا يتسبب فى ترتيب الجدول ، ولكنه يحدد الحقل المرتب الجدول على أساسه .

```
OCCURS integer-1 [TO integer-2] TIMES
    [DEPENDING ON data-name-1]
    { ASCENDING  } KEY IS data-name-2
    { DESCENDING }
    [INDEXED BY index-name-1 [index-name-2] . . .]
```

شكل ( ١٠ - ٥ )

**مثال ١٠ – ٣٦ :**

إذا كان المطلوب استرجاع وصف العمل لـ INPUT -JOB - CODE PIC X (3) من الجدول التالى:

```
01  JOB-CODES-TABLE.
    05  NUMBER-JOB-CODES          PIC S9(5)     COMP SYNC.
    05  JOB-CODES-MEANINGS        OCCURS 1 TO 100 TIMES
                                  DEPENDING ON NUMBER-JOB-CODES
                                  ASCENDING KEY IS JOB-CODE
                                  INDEXED BY JOB-CODE-INDEX.
        10  JOB-CODE              PIC X(3).
        10  JOB-DESCRIPTION       PIC X(35).
```

افرض أن الجدول مرتب فعلا ترتيباً تصاعدياً؛ طبقا لحقل JOB - CODE، ونستطيع أن نستخدم البحث الثنائى كما يلى:

```
SEARCH ALL JOB-CODES-MEANINGS
    AT END
        MOVE "*** INVALID JOB CODE ***" TO OUTPUT-JOB-DESCRIPTION
    WHEN JOB-CODE (JOB-CODE-INDEX) EQUAL INPUT-JOB-CODE
        MOVE JOB-DESCRIPTION (JOB-CODE-INDEX) TO OUTPUT-JOB-DESCRIPTION
```

## بعض الملاحظات :

(١) يجب أن يظهر اسم الجدول ، كما هو معرف بجزء OCCURS ، فى SEARCH ALL بدون دليل أو فهرس .

(٢) لا يسرى جزء VARYING مع SEARCH ALL : فتغير SEARCH ALL دائما من أول فهرس معرف للجدول الذى يجرى البحث فيه ، ولا توجد حاجه لوضع قيمة ابتدائية للفهرس عند استخدام SEARCH ALL .

(٣) لكى تعمل SEARCH ALL بطريقة صحيحة ، يجب أن يكون الجدول مرتبا فعلا ، كما هو مذكور فى جزء KEY . وإذا لم يكن الجدول مرتبا بطريقة صحيحة .. فلن تظهر رسالة تحذيرية أو رسالة خطأ ، وإنما تنتج عبارة SEARCH ALL نتائج خاطئة ببساطة ؛ خاصة إذا كان الجدول الحالى ثابت الطول ، مع عدم جزء DEPENDING ؛ فبعض العناصر البالغ؛ إجمالى عددها 100 تكون غير نشطة (نفايا) ، وهذا يجعل الجدول غير مرتب ككل ، ويتوقع أن تكون نتيجة SEARCH ALL نفايا كذلك . انظر المسألتين ٧٦ و ٧٧ من هذا الفصل ؛ لمعرفة كيف يرتب البرنامج جدولا غير مرتب .

(٤) ترتيب الأجزاء داخل OCCURS غير قابل للتغيير فى كوبل IBM OS/VS (أى إنها يجب أن تكتب بنفس الترتيب المبين فى شكل ١٠ – ٥) .

**مثال ١٠ – ٣٧ :**

يراد إدخال البيانات الموجودة فى INPUT - JOB - CODE PIC X (3) ، وفى INPUT - JOB - DESCRIPTION PIC X (35) كعنصر جديد فى جدول مثال (١٠ – ٣٦) ؛ لحفظ الجدول فى ترتيب تصاعدى طبقا للحقل JOB - CODE ، ويجب أن نبحث فى الجدول عن أول عنصر يكون أكبر من (بالنسبة لقيمة المفتاح key) العنصر الجديد ، ثم يوضع العنصر الجديد أمامه . إذا لم يوجد أى عنصر أكبر من العنصر الجديد .. فيجب وضع العنصر الجديد فى نهاية الجدول . وإذا تساوى عنصر مع العنصر الجديد الذى وجد فى الجدول .. فيتحقق شرط خطأ ، ويجب التعامل معه .

نرغب في اختبار JOB - CODE GREATER THAN ، ولكن جزء WHEN من عبارة SEARCH ALL ، مقدم فقط لـ JOB - CODE - EQUAL . وعلى هذا .. نجد أنفسنا مجبرين على استخدام SEARCH بدلا من SEARCH ALL ؛ بالرغم من أن الجدول مرتب :

```
SET JOB-CODE-INDEX TO 1
SEARCH JOB-CODES-MEANINGS
    AT END
        ADD 1 TO NUMBER-JOB-CODES
        SET JOB-CODE-INDEX TO NUMBER-JOB-CODES
        MOVE INPUT-JOB-CODE TO JOB-CODE (JOB-CODE-INDEX)
        MOVE INPUT-JOB-DESCRIPTION TO
            JOB-DESCRIPTION (JOB-CODE-INDEX)
    WHEN
        JOB-CODE (JOB-CODE-INDEX) EQUAL INPUT-JOB-CODE
            PERFORM DUPLICATE-CODE-ERROR
    WHEN

            JOB-CODE (JOB-CODE-INDEX) GREATER THAN INPUT-JOB-CODE
                ADD 1 TO NUMBER-JOB-CODES
                PERFORM MOVE-ENTRIES-UP
                    VARYING MOVE-INDEX
                        FROM NUMBER-JOB-CODES BY -1
                        UNTIL MOVE-INDEX EQUAL JOB-CODE-INDEX
                MOVE INPUT-JOB-CODE TO JOB-CODE (JOB-CODE-INDEX)
                MOVE INPUT-JOB-DESCRIPTION TO
                    JOB-DESCRIPTION (JOB-CODE-INDEX)

. . . . . . . . . . . . . . . . . . . . . . . . . . . .
MOVE-ENTRIES-UP.
    MOVE JOB-CODES-MEANINGS (MOVE-INDEX - 1) TO
        JOB-CODES-MEANINGS (MOVE-INDEX)
```

## تعليقات

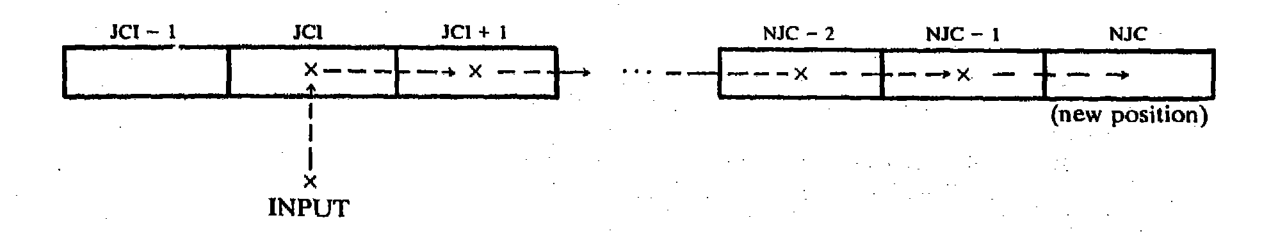
(١) تغير جزء INDEXED في شكل ( ١٠ - ٣٦ ) إلى الشكل التالي :

```
INDEXED BY JOB-CODE-INDEX
                MOVE-INDEX.
```

(٢) إذا نتج عن البحث تحقق شرط AT END .. فإننا نعرف أن العنصر الجديد أكبر من كل العناصر الموجودة . وعلى هذا .. فإننا نضيفه إلى نهاية الملف عن طريق (أ) زيادة عنصر DEPENDING (ب) نقل المعلومات الجديدة إلى آخر عنصر جديد ، نتج من الخطوة (أ) .

(٣) إذا حدد البحث عنصرا مساويا للعنصر الجديد ، فإننا ننفذ PERFORM مقطعا للخطأ .

(٤) إذا حدد البحث قيمة JOB - CODE - INDEX تعدى لها HOB - CODE القيمة INPUT - JOB - CODE .. فإننا ننتج – عند ذلك – موقعا جديدا أخيرا فى الجدول بزيادة NUMBER - JOB - CODES (انظر شكل ١٠ – ٦) . أخر عنصر فى الجدول ، عند NUMBER - JOB - CODES - 1 الآن ، ينقل إلى الموقع الجديد ، ويأخذ NUMBER - JOB - CODES - 2 مكانه ، وينقل العنصر JOB - CODE - INDEX إلى JOB - CODE - INDEX + 1 ، وتنتهى عبارة PERFORM .. تنقل بيانات المدخلات إلى JOB - CODE - INDEX . لاحظ أن الترتيب العكسى للنقل ضرورى ، وإذا حاولنا البدء بنقل البيانات من JOB - CODE INDEX للأمام ، فتكون نتيجة ذلك أن تمحو البيانات كل العناصر التالية فى الجدول .




شكل (١٠ – ٦)

## أسئلة مراجعة :

١٠ – ١ اذكر عديداً من أمثلة جداول الأعمال .

١٠ – ٢ ناقش العوامل التى تأخذ فى الاعتبار تحديد أقصى حجم لجدول .

١٠ – ٣ وضح استخدام الجداول فى تشغيل معلومات على هيئة رموز coded .

١٠ – ٤ ماذا يعنى «توسيع الرمز (الشفرة) خلال فحص الجدول» ؟

١٠ – ٥ ما الدليل وكيف يجب تعريفه ؟

١٠ – ٦ أين يمكن تعريف الجداول فى جزء البيانات ؟

١٠ – ٧ ناقش استخدام جزء VALUE فى وضع قيم ابتدائية فى جداول مخزن العمل .

١٠ – ٨ تحت أى شروط .. يعمل MOVE ZERO TO ENTIRE - TABLE بطريقة صحيحة ؟

١٠ – ٩ متى يجب استخدام عبارة MOVE فى مقطع ، ينفذ باستخدام PERFORM ؛ لوضع قيم ابتدائية لعناصر الجدول؟

١٠ – ١٠ ناقش أهمية حفظ عداد بعدد العناصر النشطة لمعظم الجداول . متى لا تكون هناك حاجة إلى مثل هذا العداد ؟

١٠ – ١١ لماذا يفضل عادة وضع قيم أولية لجدول من ملف ، بدلا من استخدام VALUE, REDEFINES ؟

١٠ – ١٢ كيف تعرف الجداول ذات البعدين ؟

١٠ – ١٣ وضح قواعد استخدام دلائل أو فهارس لمحتويات جداول ذات بعدين .

١٠ – ١٤ ناقش استخدام PERFORM .... VARYING ... AFTER مع جداول ذات بعدين .

١٠ - ١٥ وضح كيف تعمل بيانات المدخلات كدلائل . متى يمكن أن يكون هذا مفيداً ؟

١٠ - ١٦ ما أهمية التأكد من صحة حقول المدخلات المستخدمة كدلائل ؟

١٠ - ١٧ ناقش كيفية ترجمة حقول المدخلات الحرفية عددية الى قيم دلائل ؟

١٠ - ١٨ وضح معنى جزء OCCURS متداخل . ما القيد على OCCURS المتداخلة فى الكوبل ؟

١٠ - ١٩ لماذا لا يحافظ استخدام الجداول متغيرة الطول على الذاكرة الرئيسية للكمبيوتر ؟ هل تستطيع الجداول متغيرة الطول أن تحفظ مكانا فى أى مكان آخر داخل نظام الكمبيوتر ؟

١٠ - ٢٠ ما العلاقة بين السجلات متغيرة الطول والجداول متغيرة الطول ؟

١٠ - ٢١ ناقش وضع عنصر DEPENDING داخل السجل المنطقى . هل يمكن أن يتبع الجدول متغير الطول ؟

١٠ - ٢٢ اذكر القيود على جزء OCCURS ، عندما تكون الجداول متغيرة الطول مشمولة .

١٠ - ٢٣ لماذا يجب تعريف الجدول متغير الطول فى مخزن العمل ؟ هل يوفر هذا من مواقع ذاكرة الكمبيوتر ؟

١٠ - ٢٤ ناقش الأخطاء التى يمكن الوقوع فيها عند نقل جداول متغيرة الطول .

١٠ - ٢٥ لماذا لا يمكن استخدام READ ... INTO ,WRITE ... FROM مع سجلات تحتوى على جداول متغيرة الطول؟

١٠ - ٢٦ ناقش طريقتين لحذف أحد العناصر من جدول .

١٠ - ٢٧ ناقش دور رمز الحذف deletion code عند حذف محتويات منطقيا من جدول . ما آثار الحذف المنطقى على البرامج الاخرى التى تجرى تشغيلاً على الجدول ؟

١٠ - ٢٨ ناقش الخوارزمى الخاص بضغط جدول عند استخدام الحذف الواقعى (الطبيعى) .

١٠ - ٢٩ ما الفهرس ؟ متى يجب استخدام الفهارس ? كيف تعرف الفهارس ؟

١٠ - ٣٠ قارن قيم الفهرس مع قيم الدليل .

١٠ - ٣١ ما حساب العنوان address calculation ؟

١٠ - ٣٢ ما الجداول التى يجب استخدام الفهارس معها ؟

١٠ - ٣٣ اذكر ثلاث طرق يمكن فيها تعديل الفهارس ، أو وضع قيم ابتدائية لها

١٠ - ٣٤ ناقش استخدام عبارة SET . ما تكوينها ؟

١٠ - ٣٥ ناقش كيف يمكن استخدام الفهارس فى عبارة PERFORM ، و عبارة IF .

١٠ - ٣٦ ما الفهرسة النسبية ؟ ناقش استخدامها .

١٠ - ٣٧ هل هناك مكافىء للفهرسة النسبية فى الدلائل ؟

١٠ - ٣٨ وضح خوارزمى البحث المتتابع .

١٠ - ٣٩ ناقش عبارة البحث . ما تكوينها ؟ وضح .

(أ) ... VARYING

(ب) ... AT END

(جـ) ... WHEN

(د) وضع قيمة ابتدائية لفهرس SEARCH .

١٠ – ٤٠ ناقش أهمية ترتيب كتابة أجزاء WHEN فى عبارة SEARCH .

١٠ – ٤١ ما الجداول المتوازية parallel tables ؟ كيف ترتبط بجزء VARYING من عبارة SEARCH ؟

١٠ – ٤٢ ناقش كيف يمكن تطبيق SEARCH ، أو SEARCH ALL على جداول ذات بعدين .

١٠ – ٤٣ وضح خوارزمى البحث الثنائى . ما الشرط السابق الذى يجب تحقيقه لكى يعمل البحث الثنائى بطريقة صحيحة ؟

١٠ – ٤٤ ناقش الكفاءة النسبية للبحث المتتابع والبحث الثنائى . متى يجب استخدام كل منها ؟

١٠ – ٤٥ ناقش عبارة SEARCH ALL . كيف تختلف عن عبارة SEARCH ؟

١٠ – ٤٦ وضح استخدام ASCENDING KEY , DESCENDING KEY .

١٠ – ٤٧ يجب ألا تستخدم SEARCH ALL مع جدول متغير الطول ، الا اذا كان مملوءاً .لماذا ؟

١٠ – ٤٨ ناقش خوارزمى لإدخال عنصر فى جدول مرتب .

١٠ – ٤٩ ناقش خوارزمى لحذف عنصر من جدول حذفا واقعيا .

## مسائل محلولة

١٠ – ٥٠ ما الأخطاء الموجودة فى تعريفات الجداول التالية :

```
(a)  01  DECK-OF-CARDS                 PIC X(2)
                                       OCCURS 52 TIMES.

(b)  01  DECK-OF-CARDS.
         05  POKER-HAND                OCCURS 1 TO 10 TIMES
                                       DEPENDING ON NUMBER-OF-PLAYERS.
             10  CARDS                 OCCURS 5 TIMES.
                 15  RANK              PIC XX.
                 15  SUIT              PIC X.
         05  NUMBER-OF-PLAYERS         PIC S9(5)  COMP SYNC.

(c)  01  POKER-HAND.
         05  CARD                      OCCURS 5 TIMES
                                       INDEXED BY CARD-INDEX
                                       ASCENDING KEY IS RANK.
             10  RANK                  PIC XX.
             10  SUIT                  PIC X.

(d)  01  POKER-TABLE.
         05  NUMBER-OF-PLAYERS         PIC S9(4)  COMP SYNC.
         05  POKER-HAND                OCCURS 1 TO 10 TIMES
                                       DEPENDING ON NUMBER-OF-PLAYERS
                                       INDEXED BY HAND-INDEX.
             10  NUMBER-DEALT          PIC S9(4)  COMP SYNC.
             10  CARDS                 OCCURS 1 TO 7 TIMES
                                       DEPENDING ON NUMBER-DEALT
                                       INDEXED BY CARD-INDEX.
                 15  RANK              PIC XX.
                 15  SUIT              PIC X.
```

(a) يجب ألا يستخدم جزء OCCURS على المستوى 01 .

(b) هناك عنصر بيانات NUMBER - OF - PLAYERS يتبع جدول متغير الطول فى وصف السجل .

(c) INDEXED BY يجب أن يتبع ASCENDING KEY (طبقاً لكوبل IBM OS / VS ) .

(d) يتبع جزء DEPENDING ... OCCURS جزء OCCURS آخر . ومن الصحيح استخدام ما يلى :

```
10  CARDS      OCCURS 7 TIMES
               INDEXED BY CARD-INDEX.
```

أى إنه يمكن لجدول ثابت الطول أن يتبع جدول متغير الطول (أو ثابت الطول كذلك) .

١٠ – ٥١ علق على الجدول التالى :

```
01  BASEBALL-TEAM
    05  STARTING-PLAYER            OCCURS 9 TIMES
                                   INDEXED BY PLAYER-INDEX.
        10  PLAYER-NAME            PIC X(25).
        10  PLAYER-POSITION        PIC X(5).
        10  BATTING-AVERAGE        PIC SV999    COMP-3.
```

يحتوى هذا الجدول على 9 عناصر نشطة دائما . عادة ما يتغير عدد العناصر النشطة فى جدول ، سواء كان هذا أثناء تنفيذ برنامج يجرى تشغيلا على الجدول ، أو عند نقطة معينة خلال فترة استخدام البرنامج .

١٠ – ٥٢ عرف جدولا ليحتوى على عدد أسطر الكوبل التى تنتج فى الاسبوع بواسطة كل مبرمج فى إحدى المؤسسات . يعمل حاليا 55 مبرمجاً. وتخطط المؤسسة لتعين اثنى عشرمبرمجا فى العام ، ثم تعيين % 10 أكثر فى السنة التى بعدها . تقدر الفترة المتوقعة لاستخدام البرنامج الذى يجرى تشغيلا على الجدول بأربع سنوات .

```
01  PROGRAMMER-PRODUCTIVITY.
    05  NUMBER-OF-PROGRAMMERS      PIC S9(5)  COMP SYNC.
    05  PROGRAMMER-DATA            OCCURS 1 TO 94 TIMES
                                   DEPENDING ON
                                       NUMBER-OF-PROGRAMMERS
                                   INDEXED BY PROGRAMMER-INDEX.
        10  PROGRAMMER-ID          PIC X(2).
        10  LINES-OF-COBOL         PIC S9(3)  COMP-3.
```

أول سنة يستخدم فيها البرنامج .. سيكون لدى المؤسسة 70 (15+55) مبرمجا ، وفى السنة الثانية سيكون لديها 77 (70x1.10) مبرمجا ، وفى السنة الثالثة يكون لديها 85 (77x1.10) مبرمجاً ، وفى السنة الرابعة يكون لديها 94 (85x1.10) مبرمجا ، وعلى هذا .. يكون الجدول متغير الطول بحد أقصى 94 عنصرا .

١٠ – ٥٣ باستخدام جدول المسألة ٥١ .. أكتب شفرة لعرض DISPLAY الموقع ، الذى له أقل متوسط BATTING - AVER-AGE :

حيث يفيد الفهرس الثانى . عدل جزء INDEXED ليصبح INDEXED BY PLAYER - INDEX LOWEST - IN-DEX . عندئذ :

```
    SET LOWEST-INDEX TO 1
    PERFORM FIND-LOWEST-AVERAGE
        VARYING PLAYER-INDEX FROM 2 BY 1
            UNTIL PLAYER-INDEX GREATER THAN 9
    DISPLAY PLAYER-POSITION (LOWEST-INDEX)
. . . . . . . . . . . . . . . . . . . . . . . . .
FIND-LOWEST-AVERAGE.
    IF BATTING-AVERAGE (PLAYER-INDEX) LESS THAN
        BATTING-AVERAGE (LOWEST-INDEX)
            SET LOWEST-INDEX TO PLAYER-INDEX
```

١٠ – ٥٤ إذا كان ثمة قيد فى المسألة السابقة على أقل متوسط .. هل يعرض DISPLAY البرنامج أول واحد يجده ، أم يعرض آخر واحد يجده ؟

حيث إن المؤثر العلاقى LESS THAN يستخدم .. فسوف يعرض DISPLAY أول متوسط منخفض .

١٠ – ٥٥ عدل المسألة رقم ٥٣ لعرض DISPLAY آخر متوسط ، يوجد كقيد على أقل متوسط .

استخدم NOT GREATER THAN بدلاً من LESS THAN فى FIND - LOWEST - AVERAGE .

١٠ – ٥٦ حلل شفرة (3) INPUT - LOCATION - CODE PIC X إلى (30) OUTPUT - LOCATION PIC X ، مستخدماً المعلومات الموجودة فى الجدولين المتوازيين التاليين .

```
01  LOCATION-CODE-TABLE.
    05  LOCATION-CODE             PIC X(3)
                                  OCCURS 80 TIMES
                                  ASCENDING KEY
                                      LOCATION-CODE
                                  INDEXED BY CODE-INDEX.
01  LOCATION-DESCRIPTION-TABLE.
    05  LOCATION-DESCRIPTION      PIC X(30)
                                  OCCURS 80 TIMES
                                  INDEXED BY
                                      DESCRIPTION-INDEX.
```

حيث إن محتويات الجدولين متناظرة ، فإننا نستخدم (بافتراض أن LOCATION - CODE - TABLE مرتب فعلاً طبقاً لـ LOCATION - CODE ) ما يلى :

```
SEARCH ALL LOCATION-CODE
    AT END
        MOVE "** UNKNOWN LOCATION **" TO OUTPUT-LOCATION
    WHEN
        LOCATION-CODE (CODE-INDEX) EQUAL INPUT-LOCATION-CODE
            SET DESCRIPTION-INDEX TO CODE-INDEX
            MOVE LOCATION-DESCRIPTION (DESCRIPTION-INDEX)
                TO OUTPUT-LOCATION
```

١٠ – ٥٧ أعد حل المسألة السابقة إذا لم يكن الجدول مرتبا .

يجب استخدام البحث المتتابع الآن :

```
SET CODE-INDEX
     DESCRIPTION-INDEX TO 1
SEARCH LOCATION-CODE
     VARYING DESCRIPTION-INDEX
     AT END
          MOVE "*** UNKNOWN LOCATION **" TO OUTPUT-LOCATION
     WHEN
          LOCATION-CODE (CODE-INDEX) EQUAL INPUT-LOCATION-CODE
               MOVE LOCATION-DESCRIPTION (DESCRIPTION-INDEX)
                    TO OUTPUT-LOCATION
```

نظراً لأن عنصر VARYING يخص جدولا آخر .. فإنه يزداد بزيادة فهرس SEARCH (أول فهرس لجدول SEARCH) ، وعلى هذا يشير DESCRIPTION - INDEX و CODE - INDEX دائماً إلى المحتويات المناظرة .

١٠ – ٥٨ ماذا يحدث فى الكوبل اذا انخفضت قيمة الدليل أو الفهرس لأقل من 1 (أول عنصر فى الجدول) ، أو تعدت الحجم الحالى للجدول ؟

يمكن أن تثار مناطق الذاكرة التى تسبق أو تتبع الجدول . إذا كانت هناك أدنى فرصة لان تصبح قيمة الدليل أو الفهرس غير صحيحة .. فيجب أن يختبرها البرنامج قبل استخدامها فى الاتصال بالجدول .

١٠ – ٥٩ انقد ما يلى :

```
01  PROJECT-ASSIGNMENTS.
    05  PROJECT-VALUES.
        10  FILLER      PIC X(11)     VALUE "MARYX250100".
        10  FILLER      PIC X(11)     VALUE "MIKE2A70553".
        10  FILLER      PIC X(11)     VALUE "JOHNRS20050".
    05  PROGRAMMER-INFO               REDEFINES PROJECT-VALUES
                                      OCCURS 3 TIMES
                                      INDEXED BY PROGRAMMER-INDEX.
        10  FIRST-NAME                PIC X(4).
        10  PROJECT-CODE              PIC X(3).
        10  HOURS-WORKED              PIC S9(3)V9     COMP-3.
```

١ – لا يتفق جزء VALUE مع وصف محتوى الجدول ؛ إذ يفترض أن تكون HOURS - WORKED حقلاً من نوع - COMP 3 ، ولكن جزء VALUE حدد لها الاستخدام DISPLAY .

٢ – حيث يمكن أن تتغير المعلومات الموجودة فى الجدول بصفة متكررة ، فيكون استخدام REDEFINES VALUE فى وضع القيم الابتدائية .. اختبارا ضعيفا جدا . ومن الافضل كثيرا حفظ معلومات الجدول فى ملف قرص ؛ حيث يمكن تغييره ببساطة ، ويمكن نقل قيم الجدول منه كمدخلات كلما دعت الحاجة لذلك .

٣ – حيث إن عدد المبرمجين يتغير ، فيجب أن يوجد عداد مصاحب للجدول ؛ ليحدد عدد العناصر النشطة . ومثل هذا العداد يعرف عادة كعنصر على المستوى 05 تحت PROJECT - ASSIGNMENTS .

١٠ - ٦٠ صحح PROJECT - VALUES في المسألة السابقة لمواجهة الاعتراض رقم (1) في حل المسألة السابقة .

```
05  PROJECT-VALUES.
    10  FILLER  PIC X(7)       VALUE "MARYX25".
    10  FILLER  PIC S9(3)V9   COMP-3  VALUE +10.0.
    10  FILLER  PIC X(7)              VALUE "MIKE2A7".
    10  FILLER  PIC S9(3)V9   COMP-3  VALUE +55.3.
    10  FILLER  PIC X(7)              VALUE "JOHNRS2".
    10  FILLER  PIC S9(3)V9   COMP-3  VALUE +5.0.
```

١٠ - ٦١ (أ) هل ما يلي يعمل بطريقة صحيحة مع جداول المسألة ١٠ - ٥٦ ؟

```
MOVE SPACES TO LOCATION-CODE-TABLE
               LOCATION-DESCRIPTION-TABLE
```

(ب) هل ما يلي يعمل بطريقة صحيحة مع جداول المسألة ٥٢ ؟

```
MOVE SPACES TO PROGRAMMER-DATA
```

(أ) نعم ، حيث إن لكل المحتويات الوصف : PIC X (m) DISPLAY .

(ب) لا . أولا .. تتطلب PROGRAMMER - DATA دليلا أو فهرسا عند استخدامها في عبارة MOVE . ثانيا .. الحقول المجدولة هي PIC S9 (3) COMP - 3, PIC X (2) ، بحيث إنه لا يمكن لعبارة MOVE أن تضع قيما ابتدائية لهما .

١٠ - ٦٢ اكتب شفرة لوضع فراغات واصفار على التوالي لـ LINES - OF - COBOL, PROGRAMMER - ID في المسألة ١٠ - ٥٢ .

```
PERFORM CLEAR-PROGRAMMER-DATA
    VARYING PROGRAMMER-INDEX FROM 1 BY 1
    UNTIL PROGRAMMER-INDEX GREATER THAN
          NUMBER-OF-PROGRAMMERS
. . . . . . . . . . . . . . . . . . . . . . . .

CLEAR-PROGRAMMER-DATA.
    MOVE SPACES  TO  PROGRAMMER-ID (PROGRAMMER-INDEX)
    MOVE ZEROS   TO  LINES-OF-COBOL (PROGRAMMER-INDEX)
```

١٠ - ٦٣ يعرف جدولاً يمكن أن يحتوى حتى 100 عنصر من PART - ID PIC X (5) و QUANTITY PIC S9 (4) COMP لكل من : WAREHOUSE - LOCATION ، الذي له PIC X ويمكن أن يصل عدده إلى 20 .

```
01  WAREHOUSE-CONTENTS.
    05  NUMBER-LOCATIONS        PIC S9(5)  COMP SYNC.
    05  LOCATION-DATA           OCCURS 1 TO 20 TIMES
                                DEPENDING ON NUMBER-LOCATIONS
                                INDEXED BY LOCATION-INDEX.
```

```
10  WAREHOUSE-LOCATION   PIC X(3).
10  NUMBER-PARTS         PIC S9(5)  COMP.
10  PARTS-DATA           OCCURS 100 TIMES
                         INDEXED BY PART-INDEX.
    15  PART-ID          PIC X(5).
    15  QUANTITY         PIC S9(4)  COMP.
```

إلا إنه قد يكون مطلوبا تعريف PARTS - DATA على أنها OCCURS 1 TO 100 TIMES DEPENDNG ON NUMBER - PARTS فلا يمكن عمل جدول متغير الطول ، كجزء من أى جدول آخر.

١٠ – ٦٤ أى أسماء بيانات من المسألة السابقة يمكن أن يظهر فى

(أ) عبارة SEARCH ALL ؟ (ب) فى عبارة SEARCH ؟

(أ) لا شيء ، حيث إنها ليست أجزاء مفتاح تصاعدى أو تنازلى .

(ب) LOCATION - DATA أو PARTS - DATA (لاحظ أن الجدول يجب أن يكون مفهرسا لاستخدامه فى عبارات ، SEARCH ، أو SEARCH ALL

١٠ – ٦٥ اكتب كل اسم بيانات من المسألة ٦٣ ، كما يجب أن يظهر فى عبارة MOVE .

WAREHOUSE-CONTENTS; NUMBER-LOCATIONS; LOCATION-DATA (LOCATION-INDEX); WAREHOUSE-LOCATION (LOCATION-INDEX); NUMBER-PARTS (LOCATION-INDEX); PARTS-DATA (LOCATION-INDEX PART-INDEX); PART-ID (LOCATION-INDEX PART-INDEX); QUANTITY (LOCATION-INDEX PART-INDEX).

١٠ – ٦٦ بين كيف يضاف 10% إلى كل عنصر QUANTITY من جدول المسألة ٦٣ .

```
PERFORM ADD-TEN-PERCENT
    VARYING LOCATION-INDEX FROM 1 BY 1
        UNTIL LOCATION-INDEX GREATER THAN NUMBER-LOCATIONS
    AFTER PART-INDEX FROM 1 BY 1
        UNTIL PART-INDEX GREATER THAN
            NUMBER-PARTS (LOCATION-INDEX)
. . . . . . . . . . . . . . . . . . . . . . . . . .
ADD-TEN-PERCENT.
    COMPUTE QUANTITY (LOCATION-INDEX PART-INDEX) =
        QUANTITY (LOCATION-INDEX PART-INDEX) * 1.10
```

لاحظ أنه يجب فهرسة NUMBER - PARTS كما هو مبين ؛ فهى تصبح عنصرا فى الجدول .

١٠ – ٦٧ أكتب شفرة تعرض DISPLAY كل مواقع المخازن التى تخزن العنصر المعرف بواسطة INPUT - PART - ID IPC X (5) استخدم جدول المسألة ٦٣ .

```
PERFORM FIND-WHERE-PART-IS
    VARYING LOCATION-INDEX FROM 1 BY 1
    UNTIL LOCATION-INDEX GREATER THAN NUMBER-LOCATIONS
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
FIND-WHERE-PART-IS.
    SET PART-INDEX TO 1
    SEARCH PARTS-DATA
        AT END
            NEXT SENTENCE
        WHEN
            INPUT-PART-ID EQUAL
                PART-ID (LOCATION-INDEX PART-INDEX)
                    DISPLAY WAREHOUSE-LOCATION (LOCATION-INDEX)
```

١٠ – ٦٨ إذا كان معروفا السجلات المنطقية التالية :

```
FD  SHIPPING-FILE . . .
01  SHIPPING-RECORD.
    05  CUSTOMER-ID          PIC X(6).
    05  PRODUCT-ID           PIC X(4).
    05  SHIPPING-CLASS       PIC S9(2).
    05  WEIGHT-CLASS         PIC S9(2).
    05  SHIP-TO-ADDRESS      PIC X(50).
```

أفرض ان جدول تكلفة الشحن تم تعريفه وتحميله فى مخزن العمل :

```
01  SHIPPING-CHARGE-TABLE.
    05  NUMBER-SHIPPING-CLASSES      PIC S9(5)  COMP SYNC.
    05  SHIPPING-CLASS-ENTRY         OCCURS 30 TIMES
                                     INDEXED BY SHIPPING-INDEX.
        10  NUMBER-WEIGHT-CLASSES    PIC S9(5)  COMP.
        10  WEIGHT-CLASS-ENTRY       OCCURS 15 TIMES
                                     INDEXED BY WEIGHT-INDEX.
            15  SHIPPING-CHARGE      PIC S9(3)V99   COMP-3.
```

بافتراض أن SHIPPING - CLASS معرف بأنه عددى صحيح يقع بين 1 و 30 ، وأن WEIGHT - CLASS معرف بأنه عددى صحيح يقع بين 1 و 15 . أكتب شفرة تضع تكلفة الشحن المناسبة فى CUSTOMER - SHIP - CHARGE ، إذا لم يمكن تحديد موقع للتكلفة ، ضع صفراً فى CUSTOMER - SHIP - CHARGE .

```
IF SHIPPING-CLASS LESS THAN 1 OR GREATER THAN
    NUMBER-SHIPPING-CLASSES
        MOVE ZERO TO CUSTOMER-SHIP-CHARGE
ELSE IF WEIGHT-CLASS LESS THAN 1 OR GREATER THAN
    NUMBER-WEIGHT-CLASSES (SHIPPING-CLASS)
        MOVE ZERO TO CUSTOMER-SHIP-CHARGE
ELSE
    MOVE SHIPPING-CHARGE (SHIPPING-CLASS WEIGHT-CLASS)
        TO CUSTOMER-SHIP-CHARGE
```

١٠ – ٦٩ راجع المسألة ٦٨ لاستخدام فهارس بدلا من دلائل .

```
IF SHIPPING-CLASS LESS THAN 1 OR GREATER THAN
    NUMBER-SHIPPING-CLASSES
        MOVE ZERO TO CUSTOMER-SHIP-CHARGE
ELSE
    SET SHIPPING-INDEX TO SHIPPING-CLASS
    IF WEIGHT-CLASS LESS THAN 1 OR GREATER THAN
        NUMBER-WEIGHT-CLASSES (SHIPPING-INDEX)
            MOVE ZERO TO CUSTOMER-SHIP-CHARGE
    ELSE
        SET WEIGHT-INDEX TO WEIGHT-CLASS
        MOVE SHIPPING-CHARGE (SHIPPING-INDEX WEIGHT-INDEX)
            TO CUSTOMER-SHIP-CHARGE
```

١٠ – ٧٠ حدد الخطأ فيما يلى ، وحدد كيف يصحح .

```
SET A-TABLE-INDEX TO 10
SEARCH A-TABLE
    AT END NEXT SENTENCE

WHEN A-TABLE-ENTRY (A-TABLE-INDEX) EQUAL SPACES
    SEARCH ALL ANOTHER-TABLE
        WHEN ANOTHER-KEY (ANOTHER-INDEX) EQUAL ZERO
            PERFORM WHOOPEE
```

من المسموح به بدء البحث المتتابع عند أى نقطة فى الجدول . وعلى هذا بافتراض أن A - TABLE به عشرة عناصر على الاقل .. فإن عبارة SET تكون صحيحة . ولكن يجب أن يحتوى جزء WHEN على عبارة أمرية فقط ، وتصنف SEARCH ALL بأنها عبارة شرطية . ويمكن أن تكون الشفرة المناسبه كما يلى :

```
    SET A-TABLE-INDEX TO 10
    SEARCH A-TABLE
        AT END NEXT SENTENCE
        WHEN A-TABLE-ENTRY (A-TABLE-INDEX) EQUAL SPACES
            PERFORM SEARCH-ANOTHER
. . . . . . . . . . . . . . . . . . . . . . .
SEARCH-ANOTHER.
    SEARCH ALL ANOTHER-TABLE
        WHEN ANOTHER-KEY (ANOTHER-INDEX) EQUAL ZERO
            PERFORM WHOOPEE
```

١٠ – ٧١ تغيرت المسألة ٦٨ بحيث SHIPPING - CLASS أصبح له PIC X (4) كما أن WEIGHT - CLASS أصبح له PIC X (3) ، و يشمل SHIPPING - CHARGE - TABLE عنصرا على المستوى 01 ، وهو SHIPPING - CLASS - ID ، وله PIC X (4) وعنصراً على المستوى 15 ، وهو WEIGHT - CLASS - ID وله PIC X (3)

أعد كتابة الشفرة لتحويل المدخلات الحرفية عددية إلى فهارس مناسبة .

```
   SET SHIPPING-INDEX TO 1
   SEARCH SHIPPING-CLASS-ENTRY
       AT END
           MOVE ZERO TO CUSTOMER-SHIP-CHARGE
       WHEN
           SHIPPING-CLASS-ID (SHIPPING-INDEX) EQUAL
               SHIPPING-CLASS
                   PERFORM LOCATE-WEIGHT-CLASS
. . . . . . . . . . . . . . . . . . . . . . .
LOCATE-WEIGHT-CLASS.
   SET WEIGHT-INDEX TO 1
   SEARCH WEIGHT-CLASS-ENTRY
       AT END
           MOVE ZERO TO CUSTOMER-SHIP-CHARGE
       WHEN
           WEIGHT-CLASS-ID (SHIPPING-INDEX WEIGHT-INDEX)
               EQUAL WEIGHT-CLASS
                   MOVE SHIPPING-CHARGE (SHIPPING-INDEX WEIGHT-INDEX)
                       TO CUSTOMER-SHIP-CHARGE
```

١٠ – ٧٢ بمعرفة الجدول التالى (المسألة ٦٨) :

```
01  SHIPPING-TABLE.
    05  NUMBER-CLASSES        PIC S9(5)  COMP SYNC.
    05  CLASS-ENTRY           OCCURS 1 TO 80 TIMES
                              DEPENDING ON NUMBER-CLASSES
                              ASCENDING KEY CLASS-ID
                              INDEXED BY CLASS-INDEX.
        10  CLASS-ID          PIC X(4).

    10  NUMBER-WEIGHTS    PIC S9(5)  COMP.
    10  WEIGHT-ENTRY      OCCURS 15 TIMES
                          DESCENDING KEY WEIGHT-ID
                          INDEXED BY WEIGHT-INDEX.
        15  WEIGHT-ID     PIC X(3).
        15  CHARGE        PIC S9(3)V99  COMP-3.
```

راجع شفرة المسألة السابقة ؛ بحيث تستخدم بحثا ثنائيا كلما كان ذلك ممكناً مع إنهاء البحث التتابعي عندما ، يتعدى فهرس البحث عدد عناصر الجدول النشطة .

```
SEARCH ALL CLASS-ENTRY
    AT END
        MOVE ZERO TO CUSTOMER-SHIP-CHARGE
    WHEN
        CLASS-ID (CLASS-INDEX) EQUAL SHIPPING-CLASS
            PERFORM LOCATE-WEIGHT-CLASS
. . . . . . . . . . . . . . . . . . . . . . . . . .
LOCATE-WEIGHT-CLASS.
    SET WEIGHT-INDEX TO 1
    SEARCH WEIGHT-ENTRY
        AT END
            MOVE ZERO TO CUSTOMER-SHIP-CHARGE
        WHEN
            WEIGHT-INDEX GREATER THAN NUMBER-WEIGHTS (CLASS-INDEX)
                MOVE ZERO TO CUSTOMER-SHIP-CHARGE
        WHEN
            WEIGHT-ID (CLASS-INDEX WEIGHT-INDEX) EQUAL WEIGHT-CLASS
                MOVE CHARGE (CLASS-INDEX WEIGHT-INDEX)
                    TO CUSTOMER-SHIP-CHARGE
```

لاحظ أن هذا الحل أكثر كفاءة عن الحل الموجود فى المسألة السابقة .

١٠ – ٧٣ لماذا لا تستخدم SEARCH ALL فى البحث فى الجدول WEIGHT - ENTYR فى المسألة السابقة ؟ : لأنه جدول ذا طول ثابت لا يكون مملوءا دائما ، وعلى هذا لا يكون الجدول – ككل – مخزنا بالفعل .

١٠ – ٧٤ بمعرفة جداول المسألة رقم ٥٦ .. (أ) ماذاينقل فعلا بواسطة SET DESCRIPTION - INDEX TO 1 إلى ؟ DE-SCRIPTION - INDEX (ب) ماذا يضاف بواسطة SET DESCRIPTION - INDEX UP BY 1 الى DE-SCRIPTION - INDEX ؟

(أ) صفر : يمثل الفهرس الإزاحة داخل الجدول اللازمة .. للوصول إلى العنصر المطلوب . للعنصر one صفر بايت ، أبعد من بداية الجدول .

(ب) 30 : تزداد أو تقل الفهارس زيادة فى معدل الخطوات بمضاعفات طول العنصر – طول عنصر واحد لكل موقع جدول .

١٠ – ٧٥ قم بمضاهاة حسابات عناوين لغة الآلة اللازمة لعمل الدلائل مع نظيرتها اللازمة لعمل الفهارس ، لجدول ذى بعد واحد . عندما تكون K دليلا ، فإنها تكون رقم موقع العنصر .

address ENTRY (K) = address-start-of-table + (K − 1) * entry-length

وعندما تكون K فهرسا .. فإنها تكون إزاحة من بداية الجدول :

address ENTRY (K) = address-start-of-table + K

١٠ - ٧٦ ماذا يحدث إذا أريد عمل بحث ثنائى لجدول مرتب من الأصل ؟ أحد الحلول .. إذا كانت قيم الجدول مدخلات من ملف، .. فالحل عمل ترتيب إدخال insertion sort ، الذى يوضع فيه كل عنصر فى موقعه المناسب ، فى الجدول الكامل جزئيا ، كما لو كان العنصر مدخلا من الملف . طبق هذه الطريقة :

```
FD  INPUT-FILE . . .
01  INPUT-RECORD.
    05  INPUT-CODE                PIC X(3).
    05  INPUT-DESCRIPTION         PIC X(35).

01  JOB-CODES-TABLE.
    05  NUMBER-CODES              PIC S9(5)  COMP SYNC.
    05  CODE-ENTRY                OCCURS 1 TO 100 TIMES
                                  DEPENDING ON NUMBER-CODES
                                  ASCENDING KEY IS JOB-CODE
                                  INDEXED BY CODE-INDEX
                                             MOVE-INDEX.
        10  JOB-CODE              PIC X(3).
        10  JOB-DESCRIPTION       PIC X(35).

 BUILD-TABLE.
     OPEN  INPUT  INPUT-FILE
     MOVE "NO" TO END-FILE-SW
     PERFORM GET-NEXT-RECORD
     IF END-FILE-SW EQUAL "NO"
         MOVE 1 TO NUMBER-CODES
         MOVE INPUT-CODE          TO JOB-CODE (1)
         MOVE INPUT-DESCRIPTION TO JOB-DESCRIPTION (1)
         PERFORM GET-NEXT-RECORD
         PERFORM INSERT-IN-SORTED-ORDER
             UNTIL END-FILE-SW EQUAL "YES" OR NUMBER-CODES
                   EQUAL 100
     ELSE
         PERFORM FILE-EMPTY-ERROR
     .
     IF END-FILE-SW NOT EQUAL "YES"
         PERFORM TABLE-TOO-SMALL-ERROR
     .
     CLOSE INPUT-FILE
. . . . . . . . . . . . . . . . . . . . . . . . .
 INSERT-IN-SORTED-ORDER.
```

(جزء مثلما هو موجود فى مثال ١٠ - ٣٧)

١٠ - ٧٧ هناك طريقة أخرى أقل كفاءة لعمل الترتيب ، تعرف بترتيب الفقاعة bubble sort ، يقارن فيها زوج من العناصر المتجاورة .. ويحدث تبادل بينهما إذا كانا غير مرتبين . من الواضح .. إنه إذا انتقلنا خلال الجدول لعمل المقارنة والتبادل بصورة كافية .. فيصبح الجدول مرتبا . اكتب شفرة لتنفيذ ترتيب الفقاعة على جدول المسألة السابقة ، الذى يجب أن يفترض أنه أعد فعلا .

نحن نعرف فهرسا آخر للجدول SORT - INDEX ، ونعرف كذلك عنصرى بيانات : HOLD - ENTRY , HAD - TO- SWAP، لهما الشكل PIC X (3) , PIC X (38) على التوالى .

```
SET STOP-INDEX TO NUMBER-CODES
SET STOP-INDEX DOWN BY 1
MOVE "YES" TO HAD-TO-SWAP
PERFORM MAKE-A-PASS
    UNTIL STOP-INDEX NOT GREATER THAN ZERO
        OR HAD-TO-SWAP EQUAL "NO"

. . . . . . . . . . . . . . . . . . . . . . . .

MAKE-A-PASS.
    MOVE "NO" TO HAD-TO-SWAP
    PERFORM COMPARE-AND-SWAP
        VARYING CODE-INDEX FROM 1 BY 1
        UNTIL CODE-INDEX GREATER THAN STOP-INDEX
    SET STOP-INDEX DOWN BY 1
    .
. . . . . . . . . . . . . . . . . . . . . . . .

COMPARE-AND-SWAP.
    IF JOB-CODE (CODE-INDEX) GREATER THAN
        JOB-CODE (CODE-INDEX + 1)
            MOVE "YES" TO HAD-TO-SWAP
            MOVE CODE-ENTRY (CODE-INDEX) TO HOLD-ENTRY
            MOVE CODE-ENTRY (CODE-INDEX + 1) TO
                CODE-ENTRY (CODE-INDEX)
            MOVE HOLD-ENTRY TO CODE-ENTRY (CODE-INDEX + 1)
```

١٠ – ٧٨ اذكر طريقة أكثر كفاءة لتحميل الجدول فى مثال ( ١٠ – ٢١ ) .

حيث إن TABLE - VALUES - RECORD له نفس الشكل التخطيطى تماماً مثل عنصر WEIGHT - RANGE فى FREIGHT - CHARGE - TABLE .. يمكن استخدام نقل MOVE مجموعة ؛ لنقل محتويات العنصر فى عبارة واحدة . وعلى هذا .. يمكن كتابة LOAD - TABLE كما يلى (مع إلغاء الحاجة إلى MOVE FREIGHT - CHARGES ) :

```
LOAD-TABLE.
    MOVE TABLE-VALUES-RECORD TO
        WEIGHT-RANGE (NUMBER-WEIGHT-ENTRIES)
ADD 1 TO NUMBER-WEIGHT-ENTRIES
    PERFORM GET-TABLE-RECORD
```

# الفصل الحادى عشر

# تشغيل الملفات تتابعيا

# Sequential File Processing

اعتدنا الآن على : (١) إنتاج (٢) واسترجاع سجلات منطقية من ملف ، يكون تنظيمه تتابعيا -ORGANIZATION IS SEQUENTIAL ، وتعرف كذلك (٣) استخدام منطقة حالة الملف FILE STATUS في اكتشاف الأخطاء في تشغيل الملفات تتابعيا . فيما يلى توضيح أكثر لهذا ، كتوضيح أخير .

**مثال ١١ - ١ :**

فى نظم IBM يحدد رمز الحالة "00" إتماما ناجحا ، ويحدد "10" نهاية ملف ، ويحدد "30" خطأ دائماً فى نظم مكونات المدخلات أو المخرجات ، ويحدد "92" خطأ منطقيًا فى البرنامج . وعلى هذا .. فإذا سمى جزء FILE STATUS من عبارة SELECT عنصر من مخزن العمل STATUS - CODE - AREA PIC X .. يمكن أن يكون لدينا مايلى فى جزء الإجراءات .

```
OPEN   INPUT   ANY-SEQUENTIAL-FILE
IF STATUS-CODE-AREA NOT EQUAL "00"
    DISPLAY "CANNOT OPEN FILE--ABORTING RUN"
    MOVE "YES" TO PROGRAM-ABORT-SW

. . . . . . . . . . . . . . . . . . . . . . . . . . .
READ ANY-SEQUENTIAL-FILE
IF STATUS-CODE-AREA NOT EQUAL "00"
    IF STATUS-CODE-AREA EQUAL "10"
        MOVE "YES" TO END-FILE-SW
    ELSE IF STATUS-CODE-AREA EQUAL "30"
        DISPLAY "HARDWARE ERROR READING FILE"
        MOVE "YES" TO IGNORE-RECORD-SW

. . . . . . . . . . . . . . . . . . . . . . . . . . .
CLOSE ANY-SEQUENTIAL-FILE
IF STATUS-CODE-AREA NOT EQUAL "00"
    DISPLAY "CLOSE FAILED--RUN DIAGNOSTIC ON FILE"
```

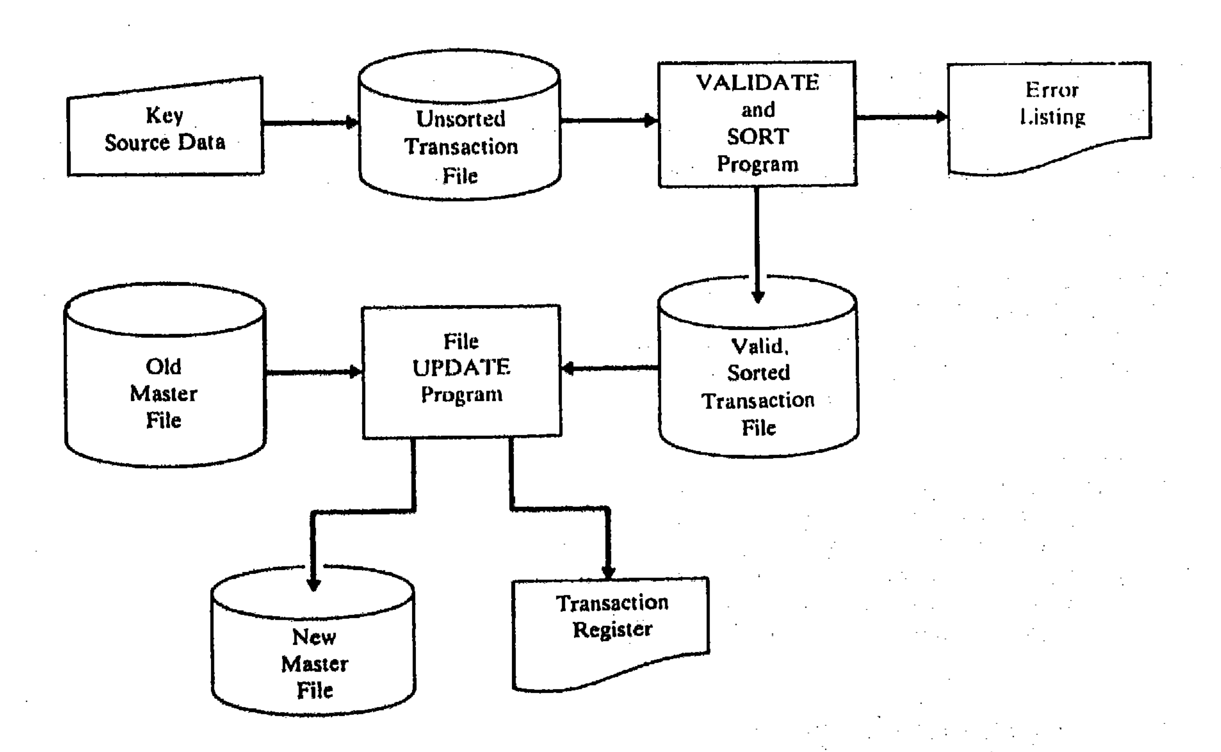
# ١١ - ١ تجديد الملفات التتابعية

تحفظ الملفات الرئيسية المرتبة تتابعيا دائما مرتبة طبقا لحقل رئيسى أو حقل مفتاح (القسم التاسع - الفصل العاشر) - وهذه حقيقة مهمة جدا عند تجديد updating الملف . وتدخل السجلات المنطقية المحتوية على معلومات مطلوب إضافتها أو حذفها من الملف الرئيسى عادة فى ملف تتابعى منفصل ؛ ملف العمليات الجارية transaction . ويحدد كل سجل عمليات جارية المفتاح لسجل الملف الرئيسى الذى يضاف أو يجرى عليه تغيير أو يحذف ، ويرتب ملف العمليات الجارية نفسه طبقا لنفس المفتاح .

ويوضح التجديد النمطى لملف تتابعى فى خريطة مسار فى شكل (١١ - ١) ، ويشتمل على الخطوات التالية :

١ - إما أن تجمع مستندات المصدر خلال فترة زمنية معينة ، ليتم إدخال الدفعة batch كلها فى ملف عمليات جارية تتابعى ، على قرص مرة واحدة ، وإما أن يتم إدخال العمليات الجارية عبر نهايات طرفية ، فى وسط الخط المفتوح كما تحدث ، إلا أن البرنامج الذى يعمل اتصالات مع النهايات الطرفية - بدلا من تشغيل هذه العمليات الجارية - يجمعها ببساطة على هيئة دفعة فى ملف عمليات جارية على قرص .

٢ - فى أى من الحالتين .. يكون لدينا ملف عمليات جارية تتابعى ، يكون قد اختبر ورتب فى نفس تتابع الملف الرئيسى المطلوب تجديده . كما يجب تصحيح العمليات الجارية الخاطئة ، والعودة إلى الخطوة الأولى ثانية .




شكل ( ١١ - ١ )

٣ – تصبح العمليات الجارية التى أجرى لها تصحيح ، ورتبت كذلك مدخلات الآن لبرنامج تجديد ملف تتابعى ، يسترجع سجلات الملف الرئيسى الموجود ، ويجرى التغييرات المحددة ، وينتج ملفاً رئيسياً جديداً new master file مجددا ، ويحفظ الملف الرئيسى القديم old master file فى هذه العملية .

٤ – يحفظ الملف الرئيسى القديم ، وملف العمليات الجارية فى مكان آمن لأغراض الاحتياط backup ؛ فإذا حدث أى شىء للملف الرئيسى الجديد .. فإنه يمكن إعادة انتاجه بتنفيذ برنامج التجديد مرة أخرى . وتعرف هذه الطريقة بطريقة الجدود grandparent والآباء parent والاطفال child للاختياطى ؛ حيث يمكن حفظ أى عدد من أجيال الملف الرئيسى القديم .

٥ – ينتج برنامج تجديد الملف التتابعى المنفذ فى الخطوة ٣ عادة تقريرا مطبوعا ، يسمى مسجل العمليات الجارية transac- tion register ؛ يسرد نتائج تطبيق كل عملية جارية على الملف الرئيسى . لاحظ أنه بالرغم من أن العملية الجارية يمكن أن تكون صحيحه فى حد ذاتها ، إلا إنها يمكن إن تكون خطأ بالنسبة للملف الرئيسى (مثل : يمكن وجود عملية جارية ، محددة بطريقة صحيحة لسجل ملف رئيسى غير موجود ) . ويجب تصحيح مثل هذه الأخطاء ، وإعادة الدورة مرة أخرى من الخطوة رقم (١) .

## ١١ - ٣ خوارزمى خط الاتزان

يخطط شكل (١١ – ٢) بشفرة شبيهة تشبة الكوبل ، وتعد إحدى الطرق الأكثر كفاءة لتوفيق السجلات فى ملف العمليات الجارية، المرتب مع نظائرها فى الملف الرئيسى المرتب ، وهى تسمى بخوارزمى خطط الاتزان balanced - line algo- rithm . قد يعالج الخوارزمى أكثر من سجل عمليات جارية واحد لكل سجل رئيسى ، وهذا موقف شائع عندما تكون العمليات الجارية على هيئة دفعات فى فترة زمنية . ويجب ملاحظة أن كل تفاصيل تشغيل حقول السجلات تترك لمقاطع ( لأجزاء ) أدنى مستوى. وعلى هذا .. يمكن استخدام أجزاء المستوى العلوى فى عديد من البرامج المختلفة لتطبيقات مختلفة .

يوجد قلب الخوارزمى فى مقطع UPDATE - MASTER - FILE ، ودائما ما يكون LOW - KEY مساويا أو أقل من TRANSACTION - KEY و OLD - MASTER - KEY ؛ فاذا كان مفتاح السجل OLD - MASTER - RECORD أقل من أو يساوى TRANSACTION - KEY (يتساوى LOW - KEY مع OLD - MASTER -KEY ) ؛ فينقل عند ذلك OLD - MASTER - RECORD الى WS - NEW - MASTER - RECORD ، وهو منطقة مخزن عمل .. تبنى فيها سجلات الملف الرئيسى الجديد . يعد MASTER - READY - FOR - OUTPUT مفتاح برنامج ، يذكر ما إذا كانت هناك بيانات نشطة فى منطقة بناء هذا السجل أم لا .. يجب أن يتحقق القارىء (بإجراء اختبار مكتبى) أنه من المستحيل تنفيذ MOVE OLD - MASTER - RECORD TO WS - NEW - MASTER - RECORD عندما يكون هناك سجل نشط فى مرحلة البناء فى منطقة WS - NEW - MASTER - RECORD . وبمجرد نقل OLD - MASTER - RECORD إلى منطقة بناء السجل الرئيسى الجديد ، يقرأ السجل الرئيسى القديم التالى فى الذاكرة الاحتياطية لمدخلات OLD - MASTER - RECORD .

```
BALANCED-LINE.
    MOVE "NO" TO MASTER-READY-FOR-OUTPUT
    OPEN INPUT   TRANSACTION-FILE
                 OLD-MASTER-FILE
         OUTPUT  NEW-MASTER-FILE
                 ERROR-REGISTER
    PERFORM GET-NEXT-TRANSACTION
    PERFORM GET-NEXT-MASTER
    PERFORM DETERMINE-LOW-KEY
    PERFORM UPDATE-MASTER-FILE
        UNTIL LOW-KEY EQUAL HIGH-VALUES
    CLOSE TRANSACTION-FILE
        OLD-MASTER-FILE
        NEW-MASTER-FILE
        ERROR-REGISTER
    STOP RUN
    .
UPDATE-MASTER-FILE.
    IF LOW-KEY EQUAL OLD-MASTER-KEY
        MOVE OLD-MASTER-RECORD TO WS-NEW-MASTER-RECORD
        MOVE "YES" TO MASTER-READY-FOR-OUTPUT
        PERFORM GET-NEXT-MASTER
    .
    IF LOW-KEY EQUAL TRANSACTION-KEY
        PERFORM PROCESS-A-TRANSACTION
            UNTIL LOW-KEY NOT EQUAL TRANSACTION-KEY
    .
    IF MASTER-READY-FOR-OUTPUT EQUAL "YES"
        WRITE NEW-MASTER-RECORD FROM WS-NEW-MASTER-RECORD
        MOVE "NO" TO MASTER-READY-FOR-OUTPUT
    .
    PERFORM DETERMINE-LOW-KEY
    .
PROCESS-A-TRANSACTION.
    IF TRANSACTION-CODE-IS-ADD
        IF MASTER-READY-FOR-OUTPUT EQUAL "YES"
            PERFORM DUPLICATE-KEY-ERROR
        ELSE
            PERFORM MOVE-TRANS-INFO-TO-WS-NEW-MASTER
            MOVE "YES" TO MASTER-READY-FOR-OUTPUT
    ELSE IF TRANSACTION-CODE-IS-CHANGE
        IF MASTER-READY-FOR-OUTPUT EQUAL "NO"
            PERFORM NO-MASTER-TO-CHANGE-ERROR
        ELSE
            PERFORM APPLY-TRANS-TO-WS-NEW-MASTER
    ELSE IF TRANSACTION-CODE-IS-DELETE
        IF MASTER-READY-FOR-OUTPUT EQUAL "NO"
            PERFORM NO-MASTER-TO-DELETE-ERROR
        ELSE
            MOVE "NO" TO MASTER-READY-FOR-OUTPUT
    ELSE
        PERFORM INVALID-TRANS-CODE-ERROR
```

شكل ( ١١ - ٢ )

```
PERFORM GET-NEXT-TRANSACTION

DETERMINE-LOW-KEY
    IF TRANSACTION-KEY IS LESS THAN OLD-MASTER-KEY
        MOVE TRANSACTION-KEY TO LOW-KEY
    ELSE
        MOVE OLD-MASTER-KEY TO LOW-KEY

GET-NEXT-TRANSACTION.
    READ TRANSACTION-FILE
        AT END
            MOVE HIGH-VALUES TO TRANSACTION-KEY

GET-NEXT-MASTER.
    READ OLD-MASTER-FILE
        AT END
            MOVE HIGH-VALUES TO OLD-MASTER-KEY
```

شكل ( ١١ - ٢ ) تكملة

الخطوة التالية .. هى تحديد إذا ما كان هناك عمليات جارية ، يجب تطبيقها على محتويات - WS - NEW - MASTER RECORD . وطالما أن TRANSACTION - KEY يستمر مساويا للقيمة الحالية LOW - KEY .. تطبق سجلات العمليات الجارية على نفس السجل الرئيسى ويتم إدخالها وتشغيلها . لاحظ أن PERFORM PROCESS - A- TRANSACTION .. UNTIL هى التى تسمح للخوارزمى بمعاملة أكثر من سجل عمليات جارية لنفس السجل الرئيسى .

بعد تنفيذ كل العمليات الجارية المناسبة ∴ يخرج الخوارزمى السجل من WS - NEW - MASTER - RECORD إلى NEW - MASTER - FILE (بافتراض وجود سجل نشط فى منطقة البناء) . وعند ذلك .. يتغير المفتاح - MASTER READY - FOR - OUTPUT ليصبح فى وضع OFF مرة أخرى .

أخيرا .. وبعد المدخلات الممكنة لسجل OLD - MASTER - RECORD ، وسجل واحد ، أو أكثر جديد من -TRANSAC TION - RECORD .. ينفذ DETERMINE - LOW - KEY مرة أخرى قبل التكرار UPDATE - MASTER - FILE .

**مثال ١١ - ٢ :**

يسجل جدول ١١ . ١ تطبيق خوارزمى خط الاتزان على ملفات اختبارية :

| | |
|---|---|
| TRANSACTION-FILE: | 15, 20, 20, 60 |
| OLD-MASTER-FILE: | 10, 20, 30, 60, 70 |

## ملاحظات خطوة بخطوة :

١ - يستدعى الخوارزمى بقراءة أول سجل من كل ملف ، وتحديد أقل مفتاح ، وتكون منطقة بناء السجل الرئيسى الجديد فارغة.

٢ - يكون LOW - KEY EQUAL OLD - MASTER - KEY صحيحا ، وعلى هذا ينسخ السجل الرئيسى القديم فى منطقة بناء السجل الرئيسى الجديد ، ويحول المفتاح إلى الحالة on ليحدد أن منطقة العمل تحتوى على سجل ، ويتم إدخال السجل الرئيسى القديم التالى .

٣ - يكون LOW - KEY EQUAL TRANSAION - KEY خطأ ، وعلى هذا .. لا يتم تشغيل أى عمليات جارية . وحيث إن MASTER - READY - FOR - OUTPUT EQUAL "YES" .. تكتب محتويات منطقة العمل (10) فى الملف الرئيسى الجديد ، ويتحول مفتاح الأعداد الى حالة off .

٤ - يقارن DETERMINE - LOW - KEY الرقم 15 مع الرقم 20 .. فيختار 15 .

٥ - بالنسبه للتنفيذ الثانى لـ UPDATE - MASTER - FILE .. لا يكون LOW - KEY مساويا لـ OLD - MASTER - KEY ، وعلى هذا .. يتم إدخال سجل رئيسى قديم . وحيث إن LOW - KEY مساو لـ TRANSACTION - KEY .. ينفذ PERFORM PROCESS - A - TRANSACTION ، لتشغيل العملية الجارية 15 . ولاحظ أنها إذا لم تكن عملية إضافة لعملية جارية .. فإنها تعامل كخطأ ؛ حيث يكون مفتاح التحديد MASTER - READY فى الوضع NO . بافتراض أنها إضافة .. تنقل العمليات الجارية إلى منطقة العمل ، ويتحول MASTER - READY الى الوضع on .

٦ - يدخل PROCESS - A - TRANSACTION الآن سجل العمليات الجارية التالى .

٧ - حيث إن LOW - KEY لا يساوى TRANSACTION - KEY .. فلن ينفذ PROCESS - A - TRANSACTION مرة أخرى فى هذا الوقت . وبدلا من ذلك يختبر .. الخوارزمى مفتاح MASTER - READY ، والذى يكون فى وضع on، وعلى هذا .. تكتب 15 فى الملف الرئيسى الجديد ، ويحول المفتاح إلى وضع off .

٨ - يقارن DETERMINE - LOW - KEY الرقم 20 مع 20 .. فيختار 20 .

٩ - عند بدء التنفيذ الثالث لـ UPDATE - MASTER - FILE .. يكون LOW - KEY مساويا OLD - MASTER - KEY، وعلى هذا .. ينقل السجل الرئيسى القدم إلى منطقة العمل ، ويوضع المفتاح فى وضع on ، ويتم إدخال السجل الرئيسى القديم التالى .

١٠ - حيث إن LOW - KEY يساوى TRANSACTION - KEY .. ينفذ PROCESS - A - TRANSACTION ، متسببا فى تشغيل أول 20 عملية جارية . ولاحظ أنها إذا كانت إضافة .. فنحن نكتشف المفتاح المكرر ، حيث تكون MASTER - READY فى وضع on . بافتراض أنه تغيير .. فإننا نجريه على السجل الرئيسى القديم 20 ، فى منطقه العمل .

١١ - يتم إدخال سجل العمليات الجارية التالى .

١٢ - حيث إن LOW - KEY لا يزال مساويا لـ TRANSACTION - KEY .. ينفذ PROCESS - A - TRANSACTION مرة أخرى ، متسببا فى تعديل السجل الرئيسى القديم 20 ، الموجود فى منطقة العمل مرة ثانية (بافتراض أن العملية الجارية هى تغيير) .

| | Execution of UPDATE-MASTER-FILE | LOW-KEY | Value in TRANSACTION-RECORD | Value in OLD-MASTER-RECORD | Value in WORKING-STORAGE NEW-MASTER Area | Ready Switch |
|---|---|---|---|---|---|---|
| 1. | first | 10 | 15 | 10 | empty | NO |
| 2. | first | 10 | 15 | 20 | 10 | YES |
| 3. | first | 10 | 15 | 20 | empty | NO |
| 4. | first | 15 | 15 | 20 | empty | NO |
| 5. | second | 15 | 15 | 20 | 15 | YES |
| 6. | second | 15 | 20 (1st) | 20 | 15 | YES |
| 7. | second | 15 | 20 (1st) | 20 | empty | NO |
| 8. | second | 20 | 20 (1st) | 20 | empty | NO |
| 9. | third | 20 | 20 (1st) | 30 | 20 | YES |
| 10. | third | 20 | 20 (1st) | 30 | 20 (changed) | YES |
| 11. | third | 20 | 20 (2nd) | 30 | 20 (changed) | YES |
| 12. | third | 20 | 20 (2nd) | 30 | 20 (changed twice) | YES |
| 13. | third | 20 | 60 | 30 | 20 (changed twice) | YES |
| 14. | third | 20 | 60 | 30 | empty | NO |
| 15. | third | 30 | 60 | 30 | empty | NO |
| 16. | fourth | 30 | 60 | 60 | 30 | YES |
| 17. | fourth | 30 | 60 | 60 | empty | NO |
| 18. | fourth | 60 | 60 | 60 | empty | NO |
| 19. | fifth | 60 | 60 | 70 | 60 | YES |
| 20. | fifth | 60 | 60 | 70 | 60 | NO |
| 21. | fifth | 60 | HIGH-VALUES | 70 | 60 | NO |
| 22. | fifth | 70 | HIGH-VALUES | 70 | 60 | NO |
| 23. | sixth | 70 | HIGH-VALUES | HIGH-VALUES | 70 | YES |
| 24. | sixth | 70 | HIGH-VALUES | HIGH-VALUES | empty | NO |
| 25. | sixth | HIGH-VALUES | HIGH-VALUES | HIGH-VALUES | empty | NO |

جدول ( ١١ - ١ )

١٣ – يتم إدخال سجل العمليات الجارية التالى .

١٤ – حيث إن LOW - KEY لم يعد مساويا TRANSACTION - KEY .. يتجه الخوارزمى للتأكد من MASTER - READY ، وحيث إنه فى حالة on ، يتم إخراج السجل المعدل الموجود فى منطقة العمل (20) إلى الملف الرئيسى الجديد، ويحول المفتاح إلى حالة off .

١٥ – يقارن DETERMINE - LOW - KEY الرقم 60 مع 30 .. فيختار 30 .

١٦ – فى بداية التنفيذ الرابع لـ UPDATE - MASTER - FILE .. يتساوى LOW - KEY مع OLD - MASTER - KEY ، بحيث ينقل السجل الرئيسى القديم إلى منطقة العمل ، ويكون المفتاح فى حالة on ، ويتم إدخال السجل الرئيسى القديم التالى .

١٧ – حيث إن LOW - KEY لا يساوى TRANSACTION - KEY .. فلن يجرى أى تشغيل على عمليات جارية ، و حيث إن MASTER - READY فى الوضع on ، فيتم إخراج محتويات منطقة العمل (السجل 30) فى الملف الرئيسى الجديد ، ويتحول MASTER - READY إلى الوضع off .

١٨ – يقارن TERMINE LOW - KEY الرقم 60 مع 60 .. فيختار 60 .

١٩ – فى بداية التنفيذ الخامس لـ UPDATE - MASTER - FILE .. يكون LOW - KEY مساويا لـ OLD - MASTER - KEY ، وعلى هذا .. ينقل السجل الرئيسى القديم إلى منطقة العمل ويصبح المفتاح فى وضع on ويتم إدخال السجل الرئيسى القديم التالى .

٢٠ – حيث إن LOW - KEY يساوى TRANSACTION - KEY .. يتم تشغيل العملية الجارية الحالية . وبافتراض أنها حذف .. يتحول MASTER - READY إلى وضع off .

٢١ – يتم إدخال العملية الجارية التالية ، ويتسبب هذا فى نهاية الملف ، الذى يضع HIGH - VALUES كقيمة لـ TRANS- ACTION - KEY ، وحيث إن LOW - KEY لا يساوى TRANSACTION - KEY.. فلن ينفذ PROCESS - A - TRANSACTION مرة أخرى . وبسبب أن MASTER - READY تحول الى الوضع off عن طريق الحذف .. فلن يتم إخراج محتويات منطقة العمل إلى الملف الرئيسى الجديد .

٢٢ – يقارن DETERMINE - LOW - KEY القيمة HIGH - VALUES مع 70 .. فيختار70 .

٢٣ – فى بداية تنفيذ UPDATE - MASTER - FILE للمرة السادسة ، يكون LOW - KEY مساويا OLD - MASTER - KEY وعلى هذا .. ينقل السجل الرئيسى القديم إلى منطقة العمل ، و يكون MASTER - READY فى وضع on ، ويتم ادخال السجل الرئيسى القديم التالى ؛ حيث يتسبب هذا فى نهاية الملف وتحديد قيمة HIGH - VALUES لـ OLD - MASTER - KEY .

٢٤ – وحيث إن LOW - KEY لايساوى TRANSACTION - KEY .. فلن ينفذ PROCESS - A - TRANSACTION. وحيث إن قيمة MASTER - READY هى "YES" يتم إخراج محتويات منطقة العمل (السجل 70) فى الملف الرئيسى الجديد ، ويتحول MASTER - READY إلى الوضع off .

٢٥ – يقارن DETERMINE - LOW - KEY القيمة HIGH - VALUES مع HIGH - VALUES ؛ مختارا HIGH - VALUES . وحيث إن LOW - KEY لا يتساوى مع HIGH - VALUES .. فلن ينفذ أى تكرار UPDATE - MAS- TER - FILE ، تكون النتيجة هى : الملف الرئيسى الجديد : السجل 10 ، السجل 15 (مضاف) ، السجل 20 (جرى عليه تغيير مرتين) ، السجل 30 ، السجل 70 .

لاحظ حذف السجل 60 .

**مثال ١١ - ٣ :**

باستخدام خوارزمى خط الاتزان ، أكتب تجديد ملف تتابعى لملف حسابات الدائنين الرئيسى التالى:

```
01  AR-CUSTOMER-MASTER-RECORD.
    05  AR-CUSTOMER-ID          PIC X(4).
    05  AR-CUSTOMER-NAME        PIC X(20).
    05  AR-NUMBER-INVOICES      PIC S9(2)        COMP-3.
05  AR-INVOICE-DATA         OCCURS 1 TO 15 TIMES
                            DEPENDING ON AR-NUMBER-INVOICES
                            ASCENDING KEY IS AR-INVOICE-DATE.
    10  AR-INVOICE-ID       PIC X(5).
    10  AR-INVOICE-DATE.
        15  AR-INVOICE-YY   PIC 99.
        15  AR-INVOICE-MM   PIC 99.
        15  AR-INVOICE-DD   PIC 99.
    10  AR-INVOICE-AMOUNT   PIC S9(5)V99   COMP-3.
```

الملف الرئيسى وملف العمليات الجارية مرتبين طبقاً لحقل ID . وهناك ثلاثة أنواع لسجلات العمليات الجارية .

- **حذف سجل عميل من الملف الرئيسى :**

| | |
|---|---|
| الأعمدة ١ - ٤ | customer ID : |
| العمود ٥ | رمز العملية الجارية "1" : |
| الأعمدة ٦ - ٨٠ | غير مستخدمة : |

- **اضافة سجل عميل جديد للملف الرئيسى :**

| | |
|---|---|
| الأعمدة ١ - ٤ | customer ID : |
| العمود ٥ | رمز العملية الجارية "2" : |
| الأعمدة ٦ - ٢٥ | اسم العميل : |
| الأعمدة ٢٦ - ٣٠ | invoice ID : |
| الأعمدة ٣١ - ٣٦ | invoice date (yymmdd) : |
| الأعمدة ٣٧ - ٤٣ | amount + (DISPLAY) : |
| الأعمدة ٤٤ - ٨٨ | غير مستخدم : |

- **تغيير بيانات الفاتورة الموجودة :**

| | |
|---|---|
| الأعمدة ١١ - ٤ | Customer ID : |
| العمود ٥ | رمز الفاتورة ( "I" أو "P" أو "A") : |
| الأعمدة ٧ - ١١ | invoice ID : |

الأعمدة ١٢ – ١٧ invoice date (yymmdd) :

الأعمدة ١٨ – ٢٤ amount [S9(5)V99 DISPLAY] :

الأعمدة ٢٥ – ٨٠ : غير مستخدم

يوجه الرمز "I" الانتباه الى وجود محتوى جديد ، يراد إدخاله فى جدول بيانات الفاتورة (مع حفظ الجدول مرتبا طبقا لتاريخ الفاتورة) . يدعو الرمز "A" إلى استبدال الكمية الموجودة فى الجدول ، بالكمية الموجودة فى سجل العمليات الجارية . يعطى الرمز «I» إشارة بأن هناك مبلغاً مدفوعاً بقيمة الفاتورة . إذا كانت قيمة الفاتورة صفراً بعد طرح هذا المبلغ .. تحذف محتويات بيانات الفاتورة واقعيا ، إما إذا كان سالبا .. يسرى الحذف مع كتابة سجل فى ملف تتابعى اسمه CREDIT - FILE ، وتوجد فيه الحقول INVOICE - ID , CUSTOMER - NAME, CUSTOMER - ID , CREDIT - BAL-, ANCE [S9 (5)V99COMP - 3] , INVOICE - DATE (yymmdd) . إذا ترك سجل رئيسى بدون فواتير ، .. فإنه يحذف من الملف . بالإضافة إلى CREDIT - FILE, NEW - AR - CUSTOMER - MASTER - FILE .. يتم إخراج ERROR - REPORT ، يشمل عناوين للصفحة (مع رقم الصفحة وتاريخ التنفيذ) لكل صفحة ، وكل حقول العمليات الجارية لكل سجل خطأ وعبارة باللغة الإنجليزية تصف الخطأ .. يجب أن تغطى الأخطاء التالية :

(١) رمز عملية جارية 1 ، مع عدم وجود customer ID فى الملف .

(٢) رمز عملية جارية 2 ، مع وجود customer ID فعلا فى الملف .

(٣) رمز عملية جارية 3 ، مع عدم وجود customer ID فى الملف .

(٤) رمز العملية الجارية 3I ، مع وجود invoice ID فعلا فى جدول الفواتير .

(٥) رمز العملية 3A ، مع عدم وجود invoice ID فى جدول الفواتير .

(٦) رمز العملية 3P ، مع عدم وجود invoice ID فى جدول الفواتير ، أو مع عدم اتفاق تاريخ الفاتورة ، مع تاريخ الفاتورة فى الملف الرئيسى .

(٧) لا يكون تاريخ فاتورة العملية الجارية أو كمية الفاتورة عدديا .

يوجد أحد الحلول فى شكل (١١ – ٣) فيما يلى .

```
          IDENTIFICATION DIVISION.

          PROGRAM-ID.  SEQUPDAT.

          AUTHOR.  LARRY NEWCOMER.
          INSTALLATION. PENN STATE UNIVERSITY, YORK CAMPUS.

          DATE-WRITTEN.   MAY 1983.
          DATE-COMPILED.  MAY  9,1983.
          SECURITY. NONE.

         *    SEQUENTIAL FILE UPDATE USING THE BALANCED LINE
         *    ALGORITHM.  INPUT:  OLD MASTER FILE
```

شكل (١١ – ٣)

```
      *                              TRANSACTION FILE
      *                      OUTPUT: NEW MASTER FILE
      *                              ERROR REPORT
      *                              CREDIT FILE

       ENVIRONMENT DIVISION.

       CONFIGURATION SECTION.
       SOURCE-COMPUTER.   IBM-3081.
       OBJECT-COMPUTER.   IBM-3081.

       INPUT-OUTPUT SECTION.

       FILE-CONTROL.

           SELECT OLD-MASTER-FILE
               ASSIGN TO OLDMAST
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
               .

           SELECT TRANSACTION-FILE
               ASSIGN TO TRANS
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
               .

           SELECT NEW-MASTER-FILE
               ASSIGN TO NEWMAST
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
               .

           SELECT CREDIT-FILE
               ASSIGN TO CREDITS
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
               .

           SELECT ERROR-REPORT
               ASSIGN TO ERRORLOG
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
               .

       DATA DIVISION.

       FILE SECTION.

      *
      * ------------------------------------------------------------
      *
      *    USE OF VARIABLE-LENGTH RECORDS TO SAVE DISK/TAPE SPACE
      *
      * ------------------------------------------------------------
      *
       FD  OLD-MASTER-FILE
           BLOCK CONTAINS 0 RECORDS
```

شكل ( ١١ - ٣ ) تكملة

```
         RECORD CONTAINS 41 TO 251 CHARACTERS
         LABEL RECORDS ARE STANDARD
         .

     01  OM-AR-CUSTOMER-RECORD.
         05  OM-AR-CUSTOMER-ID          PIC X(4).
         05  OM-AR-CUSTOMER-NAME        PIC X(20).
         05  OM-AR-NUMBER-INVOICES      PIC S9(2)      COMP-3.
    *
    * ----------------------------------------------------------
    *
    *    VARIABLE-LENGTH TABLE DEFINED WITHIN LOGICAL RECORD
    *
    * ----------------------------------------------------------
    *
         05  OM-AR-INVOICE-DATA         OCCURS 1 TO 15 TIMES
                                        DEPENDING ON
                                            OM-AR-NUMBER-INVOICES
                                        .
             10  OM-INVOICE-ID          PIC X(5).
             10  OM-INVOICE-DATE.
                 15  OM-INVOICE-YY      PIC 99.
                 15  OM-INVOICE-MM      PIC 99.
                 15  OM-INVOICE-DD      PIC 99.
             10  OM-INVOICE-AMOUNT      PIC S9(5)V99   COMP-3.

     FD  TRANSACTION-FILE
         RECORD CONTAINS 80 CHARACTERS
         LABEL RECORDS ARE OMITTED
         .

     01  TRANSACTION-RECORD.
         05  TRANS-SORT-KEY.
             10  TRANS-CUSTOMER-ID      PIC X(4).
             10  TRANS-CODE             PIC X.
                 88  DELETION-TRANSACTION               VALUE "1".
                 88  ADD-TRANSACTION                    VALUE "2".
                 88  CHANGE-TRANSACTION                 VALUE "3".
         05  TRANS-VARIABLE-AREA        PIC X(75).
    *
    * ----------------------------------------------------------
    *
    *    USE OF REDEFINES TO DESCRIBE THE VARIOUS FORMS OF
    *    TRANSACTION RECORDS
    *
    * ----------------------------------------------------------
    *
         05  TRANS-ADD-AREA             REDEFINES TRANS-VARIABLE-AREA.
             10  ADD-CUSTOMER-NAME      PIC X(20).
             10  ADD-INVOICE-ID         PIC X(5).
             10  ADD-INVOICE-DATE       PIC 9(6).
             10  ADD-AMOUNT             PIC S9(5)V99.
             10  FILLER                 PIC X(37).
         05  TRANS-CHANGE-AREA          REDEFINES TRANS-VARIABLE-AREA.
             10  TRANS-INVOICE-CODE     PIC X.
                 88  INSERTION-CODE                     VALUE "I".
                 88  ADJUSTMENT-CODE                    VALUE "A".
```

شكل ( ١١ - ٣ ) تكملة

```
                    88  PAYMENT-CODE                         VALUE "P".
                10  CHANGE-INVOICE-ID      PIC X(5).
                10  CHANGE-INVOICE-DATE    PIC X(6).
                10  CHANGE-AMOUNT          PIC S9(5)V99.
                10  CHANGE-AMOUNT-X        REDEFINES CHANGE-AMOUNT
                                           PIC X(7).
                10  FILLER                 PIC X(56).


        FD  NEW-MASTER-FILE
            BLOCK CONTAINS 0 RECORDS
            RECORD CONTAINS 41 TO 251 CHARACTERS
            LABEL RECORDS ARE STANDARD
            .

        01  NM-AR-CUSTOMER-RECORD.
            05  NM-AR-CUSTOMER-ID          PIC X(4).
            05  NM-AR-CUSTOMER-NAME        PIC X(20).
            05  NM-AR-NUMBER-INVOICES      PIC S9(2)       COMP-3.
            05  NM-AR-INVOICE-DATA         OCCURS 1 TO 15 TIMES
                                           DEPENDING ON
                                               NM-AR-NUMBER-INVOICES
                                           .
                10  NM-INVOICE-ID          PIC X(5).
                10  NM-INVOICE-DATE.
                    15  NM-INVOICE-YY      PIC 99.
                    15  NM-INVOICE-MM      PIC 99.
                    15  NM-INVOICE-DD      PIC 99.
                10  NM-INVOICE-AMOUNT      PIC S9(5)V99    COMP-3.

        FD  CREDIT-FILE
            BLOCK CONTAINS 0 RECORDS
            RECORD CONTAINS 39 CHARACTERS
            LABEL RECORDS ARE STANDARD
            .

        01  CREDIT-BALANCE-RECORD.
            05  CREDIT-CUSTOMER-ID         PIC X(4).
            05  CREDIT-CUSTOMER-NAME       PIC X(20).
            05  CREDIT-INVOICE-ID          PIC X(5).
            05  CREDIT-INVOICE-DATE        PIC 9(6).
            05  CREDIT-BALANCE             PIC 9(5)V99     COMP-3.

        FD  ERROR-REPORT
            RECORD CONTAINS 132 CHARACTERS
            LABEL RECORDS ARE OMITTED
            LINAGE IS 60
                WITH FOOTING AT 59
                LINES AT TOP 3
                LINES AT BOTTOM 3
            .

        01  ERROR-REPORT-LINE              PIC X(132).

        WORKING-STORAGE SECTION.

        01  WS-SWITCHES-AND-WORK-AREAS.
            05  WS-PAGE-NUMBER             PIC S9(3)       COMP-3.
            05  LOW-KEY                    PIC X(4).
```

شكل ( ١١ - ٣ ) تكملة

```
           05  MAXIMUM-TABLE-SIZE        PIC S9(3)         COMP-3
                                                           VALUE +15.
           05  MASTER-READY              PIC X(3).
               88  MASTER-RECORD-PRESENT                   VALUE "YES".
               88  NO-MASTER-RECORD                        VALUE "NO ".
           05  SYSTEM-DATE.
               10  SYSTEM-YY             PIC 99.
               10  SYSTEM-MM             PIC 99.
               10  SYSTEM-DD             PIC 99.
           05  WS-DIFFERENCE             PIC S9(5)V99     COMP-3.
      *
      *
      * -----------------------------------------------------------
      *
      *    ERROR MESSAGE TEXT DEFINED AS PROGRAM CONSTANTS RATHER
      *    THAN USING NON-NUMERIC LITERALS IN PROCEDURE DIVISION
      *
      * -----------------------------------------------------------
      *
       01  WS-ERROR-MESSAGE-AREAS.
           05  ADD-ERROR-MESSAGE          PIC X(42)
                                          VALUE "DUPLICATE KEY--NO ADD".
           05  CHANGE-ERROR-MESSAGE       PIC X(42)
                                          VALUE "KEY NOT FOUND--NO CHANGE".
           05  DELETION-ERROR-MESSAGE     PIC X(42)
                                          VALUE "KEY NOT FOUND--NO DELETE".
           05  INVALID-TRANS-CODE-MESSAGE
                                          PIC X(42)
                                          VALUE "INVALID CODE--NO ACTION".
           05  INVALID-CHANGE-CODE-MESSAGE
                                          PIC X(42)
                                          VALUE "INVALID CHANGE--IGNORED".
           05  ADJUST-ERROR-MESSAGE       PIC X(42)
                                          VALUE "INVOICE NOT FOUND".
           05  INVALID-FIELDS-MESSAGE     PIC X(42)
                                          VALUE "INVALID AMOUNT/DATE".
           05  NO-PAYMENT-INVOICE-MESSAGE
                                          PIC X(42)
                                VALUE "NO INVOICE--PAYMENT NOT CREDITED".
           05  DUPLICATE-INVOICE-MESSAGE
                                          PIC X(42)
                                VALUE "DUPLICATE INVOICE--IGNORED".
           05  TABLE-OVERFLOW-MESSAGE     PIC X(42)
                                          VALUE ">15 INVOICES--IGNORED".

      *
      * -----------------------------------------------------------
      *
      *    WORKING-STORAGE AREA USED IN BALANCED LINE ALGORITHM
      *    TO CONSTRUCT NEW-MASTER-FILE RECORD WITH ALL
      *    TRANSACTIONS APPLIED
      *
      * -----------------------------------------------------------
      *
       01  WS-AR-CUSTOMER-RECORD.
           05  WS-AR-CUSTOMER-ID         PIC X(4).
           05  WS-AR-CUSTOMER-NAME       PIC X(20).
           05  WS-AR-NUMBER-INVOICES     PIC S9(2)        COMP-3.
           05  WS-AR-INVOICE-DATA        OCCURS 1 TO 15 TIMES
```

شكل ( ١١-٣ ) تكملة

```
                                      DEPENDING ON
                                         WS-AR-NUMBER-INVOICES
                                      ASCENDING KEY IS
                                         WS-INVOICE-DATE
                                      INDEXED BY WS-INDEX
                                                 WS-INDEX-2
                                      .
              10  WS-INVOICE-ID       PIC X(5).
              10  WS-INVOICE-DATE.
                  15  WS-INVOICE-YY   PIC 99.
                  15  WS-INVOICE-MM   PIC 99.
                  15  WS-INVOICE-DD   PIC 99.
              10  WS-INVOICE-AMOUNT   PIC S9(5)V99   COMP-3.


      01  WS-ERROR-PAGE-TITLE.
          05  FILLER                  PIC X(20)      VALUE SPACES.
          05  FILLER                  PIC X(25)
                                      VALUE "TRANSACTION ERROR LOG".
          05  FILLER                  PIC X(5)       VALUE "PAGE ".
          05  WS-ERROR-PAGE-NUMBER    PIC ZZ9.
          05  FILLER                  PIC XX         VALUE SPACES.
          05  WS-ERROR-PAGE-DATE.
              10  WS-ERROR-MM         PIC Z9.
              10  FILLER              PIC X          VALUE "/".
              10  WS-ERROR-DD         PIC 99.
              10  FILLER              PIC X          VALUE "/".

              10  WS-ERROR-YY         PIC 99.
          05  FILLER                  PIC X(69)      VALUE SPACES.

      01  WS-ERROR-HEADING.
          05  FILLER                  PIC X(9)       VALUE "CUSTOMER"
          05  FILLER                  PIC X(5)       VALUE "CODE".
          05  FILLER                  PIC X(118)
                                      VALUE "REST OF TRANSACTION".

      01  WS-ERROR-LINE.
          05  WS-ERROR-CUSTOMER-ID    PIC X(4).
          05  FILLER                  PIC X(5)       VALUE SPACES.
          05  WS-ERROR-CODE           PIC X.
          05  FILLER                  PIC X(4)       VALUE SPACES.
          05  WS-ERROR-REST-OF-TRAN   PIC X(75).
          05  FILLER                  PIC X(1)       VALUE SPACES.
          05  WS-ERROR-MESSAGE        PIC X(42).


      PROCEDURE DIVISION.

     *
     * -----------------------------------------------------------
     *
     *    COBOL IMPLEMENTATION OF BALANCED LINE ALGORITHM*
     *    FOR SEQUENTIAL FILE UPDATE
     *
     * -----------------------------------------------------------
     *
      000-SEQUENTIAL-FILE-UPDATE.

```

شكل ( ١١ - ٣ ) تكملة

```
           MOVE "NO" TO MASTER-READY
           MOVE ZERO TO WS-PAGE-NUMBER
           ACCEPT SYSTEM-DATE FROM DATE
           MOVE SYSTEM-YY TO WS-ERROR-YY
           MOVE SYSTEM-MM TO WS-ERROR-MM
           MOVE SYSTEM-DD TO WS-ERROR-DD
           OPEN    INPUT                    OLD-MASTER-FILE
                                            TRANSACTION-FILE
                   OUTPUT                   NEW-MASTER-FILE
                                            CREDIT-FILE
                                            ERROR-REPORT
           PERFORM 170-PRINT-ERROR-REPORT-HEADING
           PERFORM 030-READ-TRANSACTION
           PERFORM 040-READ-MASTER
           PERFORM 050-DETERMINE-LOW-KEY
           PERFORM 010-UPDATE-MASTER-FILE
               UNTIL LOW-KEY EQUAL HIGH-VALUES

           CLOSE   OLD-MASTER-FILE
                   TRANSACTION-FILE
                   NEW-MASTER-FILE
                   CREDIT-FILE
                   ERROR-REPORT
           STOP RUN
           .

       010-UPDATE-MASTER-FILE.

           IF LOW-KEY EQUAL OM-AR-CUSTOMER-ID
      *
      * -----------------------------------------------------------
      *
      *    ILLUSTRATES MOVING VARIABLE-LENGTH TABLE
      *
      * -----------------------------------------------------------
      *
               MOVE OM-AR-NUMBER-INVOICES TO WS-AR-NUMBER-INVOICES
               MOVE OM-AR-CUSTOMER-RECORD TO WS-AR-CUSTOMER-RECORD
               MOVE "YES" TO MASTER-READY
               PERFORM 040-READ-MASTER
           .
           IF LOW-KEY EQUAL TRANS-CUSTOMER-ID
               PERFORM 020-PROCESS-TRANSACTIONS
                   UNTIL LOW-KEY NOT EQUAL TRANS-CUSTOMER-ID
           .
           IF MASTER-RECORD-PRESENT
               PERFORM 060-WRITE-NEW-MASTER-RECORD
               MOVE "NO" TO MASTER-READY
           .
           PERFORM 050-DETERMINE-LOW-KEY
           .

       020-PROCESS-TRANSACTIONS.

      *
      * -----------------------------------------------------------
      *
      *    NOTE THAT THIS VERSION OF THE BALANCED LINE ALGORITHM
      *    VALIDATES ALL TRANSACTIONS BEFORE APPLYING THEM
      *
      * -----------------------------------------------------------
      *
```

شكل ( ١١ - ٣ ) تكملة

```
     IF ADD-TRANSACTION
         IF MASTER-RECORD-PRESENT
             MOVE ADD-ERROR-MESSAGE TO WS-ERROR-MESSAGE
             PERFORM 070-WRITE-ERROR-LINE
         ELSE
             IF ADD-INVOICE-DATE NUMERIC AND ADD-AMOUNT NUMERIC
                  PERFORM 080-MOVE-TRANS-TO-MASTER-AREA
                  MOVE "YES" TO MASTER-READY
             ELSE
                  MOVE INVALID-FIELDS-MESSAGE TO WS-ERROR-MESSAGE
                  PERFORM 070-WRITE-ERROR-LINE
     ELSE IF CHANGE-TRANSACTION
         IF NO-MASTER-RECORD
             MOVE CHANGE-ERROR-MESSAGE TO WS-ERROR-MESSAGE
             PERFORM 070-WRITE-ERROR-LINE
         ELSE
             IF      CHANGE-INVOICE-DATE NUMERIC
                 AND CHANGE-AMOUNT NUMERIC
                  PERFORM APPLY-TRANSACTION-TO-MASTER
             ELSE
                  MOVE INVALID-FIELDS-MESSAGE TO WS-ERROR-MESSAGE
                  PERFORM 070-WRITE-ERROR-LINE
     ELSE IF DELETION-TRANSACTION
         IF NO-MASTER-RECORD
             MOVE DELETION-ERROR-MESSAGE TO WS-ERROR-MESSAGE
             PERFORM 070-WRITE-ERROR-LINE
         ELSE
             MOVE "NO" TO MASTER-READY
     ELSE
         MOVE INVALID-TRANS-CODE-MESSAGE TO WS-ERROR-MESSAGE
         PERFORM 070-WRITE-ERROR-LINE
     .
     PERFORM 030-READ-TRANSACTION
     .

 030-READ-TRANSACTION.

     READ TRANSACTION-FILE
         AT END
             MOVE HIGH-VALUES TO TRANS-CUSTOMER-ID
     .

 040-READ-MASTER.

     READ OLD-MASTER-FILE
         AT END
             MOVE HIGH-VALUES TO OM-AR-CUSTOMER-ID
     .

 050-DETERMINE-LOW-KEY.

     IF TRANS-CUSTOMER-ID IS LESS THAN OM-AR-CUSTOMER-ID
         MOVE TRANS-CUSTOMER-ID TO LOW-KEY
     ELSE
         MOVE OM-AR-CUSTOMER-ID TO LOW-KEY
     .

 060-WRITE-NEW-MASTER-RECORD.
```

شكل ( ١١ - ٣ ) تكملة

```
      *
      *-------------------------------------------------------------
      *
      *    CHECK FOR VARIABLE-LENGTH DEPENDING ON ITEM = ZERO
      *    BEFORE WRITING LOGICAL RECORD TO FILE
      *
      *-------------------------------------------------------------
      *
           IF WS-AR-NUMBER-INVOICES POSITIVE
               MOVE WS-AR-NUMBER-INVOICES TO NM-AR-NUMBER-INVOICES
               MOVE WS-AR-CUSTOMER-RECORD TO NM-AR-CUSTOMER-RECORD
               WRITE NM-AR-CUSTOMER-RECORD
           .

       070-WRITE-ERROR-LINE.

           MOVE TRANS-CUSTOMER-ID        TO WS-ERROR-CUSTOMER-ID
           MOVE TRANS-CODE               TO WS-ERROR-CODE
           MOVE TRANS-VARIABLE-AREA      TO WS-ERROR-REST-OF-TRAN
           WRITE ERROR-REPORT-LINE
               FROM WS-ERROR-LINE
               AFTER ADVANCING 2 LINES
               AT END-OF-PAGE
                   PERFORM 170-PRINT-ERROR-REPORT-HEADING
           .

       080-MOVE-TRANS-TO-MASTER-AREA.

           MOVE TRANS-CUSTOMER-ID    TO WS-AR-CUSTOMER-ID
           MOVE ADD-CUSTOMER-NAME    TO WS-AR-CUSTOMER-NAME
           MOVE 1                    TO WS-AR-NUMBER-INVOICES
           MOVE ADD-INVOICE-ID       TO WS-INVOICE-ID (1)
           MOVE ADD-INVOICE-DATE     TO WS-INVOICE-DATE (1)
           MOVE ADD-AMOUNT           TO WS-INVOICE-AMOUNT (1)
           .

       APPLY-TRANSACTION-TO-MASTER.

           IF INSERTION-CODE
               PERFORM 090-INSERT-NEW-INVOICE
           ELSE IF ADJUSTMENT-CODE
               PERFORM 130-ADJUST-AN-INVOICE
           ELSE IF PAYMENT-CODE
               PERFORM 140-APPLY-PAYMENT
           ELSE
               MOVE INVALID-CHANGE-CODE-MESSAGE TO WS-ERROR-MESSAGE
               PERFORM 070-WRITE-ERROR-LINE


       090-INSERT-NEW-INVOICE.

      *
      *-------------------------------------------------------------
      *
      *    CHECK FOR TABLE OVERFLOW BEFORE INSERTING NEW INVOICE
      *
      *-------------------------------------------------------------
      *
```

شكل ( ١١ - ٣ ) تكملة

```
           IF WS-AR-NUMBER-INVOICES EQUAL MAXIMUM-TABLE-SIZE
               MOVE TABLE-OVERFLOW-MESSAGE TO WS-ERROR-MESSAGE
               PERFORM 070-WRITE-ERROR-LINE
           ELSE
      *
      * -----------------------------------------------------------
      *
      *    ILLUSTRATES USE OF SEQUENTIAL SEARCH
      *
      * -----------------------------------------------------------
      *
               SET WS-INDEX TO 1
               SEARCH WS-AR-INVOICE-DATA
                   AT END
                       PERFORM 100-LOCATE-AND-MOVE
                   WHEN WS-INVOICE-ID (WS-INDEX) EQUAL CHANGE-INVOICE-ID
                      MOVE DUPLICATE-INVOICE-MESSAGE TO WS-ERROR-MESSAGE
                      PERFORM 070-WRITE-ERROR-LINE
           .

       100-LOCATE-AND-MOVE.

           SET WS-INDEX TO 1
           SEARCH WS-AR-INVOICE-DATA
               AT END
                   ADD 1 TO WS-AR-NUMBER-INVOICES
                   PERFORM 120-MOVE-IN-NEW-ENTRY
               WHEN WS-INVOICE-DATE (WS-INDEX) GREATER THAN
                   CHANGE-INVOICE-DATE
                       ADD 1 TO WS-AR-NUMBER-INVOICES
                       PERFORM 110-MOVE-ENTRIES-DOWN
                           VARYING WS-INDEX-2 FROM WS-AR-NUMBER-INVOICES
                           BY -1
                           UNTIL WS-INDEX-2 EQUAL WS-INDEX
                       PERFORM 120-MOVE-IN-NEW-ENTRY
           .

       110-MOVE-ENTRIES-DOWN.

           MOVE WS-AR-INVOICE-DATA (WS-INDEX-2 - 1) TO
                WS-AR-INVOICE-DATA (WS-INDEX-2)
           .

       120-MOVE-IN-NEW-ENTRY.

           MOVE CHANGE-INVOICE-ID   TO WS-INVOICE-ID (WS-INDEX)
           MOVE CHANGE-INVOICE-DATE TO WS-INVOICE-DATE (WS-INDEX)
           MOVE CHANGE-AMOUNT       TO WS-INVOICE-AMOUNT (WS-INDEX)
           .

       130-ADJUST-AN-INVOICE.

           SET WS-INDEX TO 1
           SEARCH WS-AR-INVOICE-DATA
               AT END
                   MOVE ADJUST-ERROR-MESSAGE TO WS-ERROR-MESSAGE
                   PERFORM 070-WRITE-ERROR-LINE
               WHEN WS-INVOICE-ID (WS-INDEX) EQUAL CHANGE-INVOICE-ID
```

شكل ( ١١ - ٣ ) تكملة

```
                 MOVE CHANGE-AMOUNT TO WS-INVOICE-AMOUNT (WS-INDEX)


       140-APPLY-PAYMENT.

           SET WS-INDEX TO 1
           SEARCH WS-AR-INVOICE-DATA
               AT END
                   MOVE NO-PAYMENT-INVOICE-MESSAGE TO WS-ERROR-MESSAGE
                   PERFORM 070-WRITE-ERROR-LINE
               WHEN WS-INVOICE-ID (WS-INDEX) EQUAL CHANGE-INVOICE-ID
                   PERFORM 150-CALCULATE-BALANCE


       150-CALCULATE-BALANCE.

           SUBTRACT CHANGE-AMOUNT FROM WS-INVOICE-AMOUNT (WS-INDEX)
               GIVING WS-INVOICE-AMOUNT (WS-INDEX)
                      WS-DIFFERENCE
      *
      * ----------------------------------------------------------
      *
      *    IF OVERPAID, WRITE CUSTOMER INFORMATION TO CREDIT FILE
      *
      * ----------------------------------------------------------
      *
           IF WS-DIFFERENCE NEGATIVE
               MOVE TRANS-CUSTOMER-ID      TO CREDIT-CUSTOMER-ID
               MOVE WS-AR-CUSTOMER-NAME    TO CREDIT-CUSTOMER-NAME
               MOVE CHANGE-INVOICE-ID      TO CREDIT-INVOICE-ID
               MOVE CHANGE-INVOICE-DATE    TO CREDIT-INVOICE-DATE
               MOVE WS-DIFFERENCE          TO CREDIT-BALANCE
           *   WRITE CREDIT-BALANCE-RECORD

      *
      * ----------------------------------------------------------
      *
      *    IF OVERPAID OR PAID OUT, REMOVE INVOICE FROM TABLE
      *
      * ----------------------------------------------------------
      *
           IF WS-DIFFERENCE NOT POSITIVE
               PERFORM 160-REMOVE-INVOICE
                   VARYING WS-INDEX FROM WS-INDEX BY 1
                   UNTIL WS-INDEX GREATER THAN WS-AR-NUMBER-INVOICES
               SUBTRACT 1 FROM WS-AR-NUMBER-INVOICES
           .

       160-REMOVE-INVOICE.

           MOVE WS-AR-INVOICE-DATA (WS-INDEX + 1) TO
                WS-AR-INVOICE-DATA (WS-INDEX)
           .

       170-PRINT-ERROR-REPORT-HEADING.

           ADD 1 TO WS-PAGE-NUMBER
           MOVE WS-PAGE-NUMBER TO WS-ERROR-PAGE-NUMBER
           WRITE ERROR-REPORT-LINE
```

شكل (١١ - ٣) تكملة

```
          FROM WS-ERROR-PAGE-TITLE
          AFTER ADVANCING PAGE
      WRITE ERROR-REPORT-LINE
          FROM WS-ERROR-HEADING
          AFTER ADVANCING 2 LINES
```

شكل (١١ - ٣) تكملة

## ١١ - ٣ التجديد فى نفس المكان

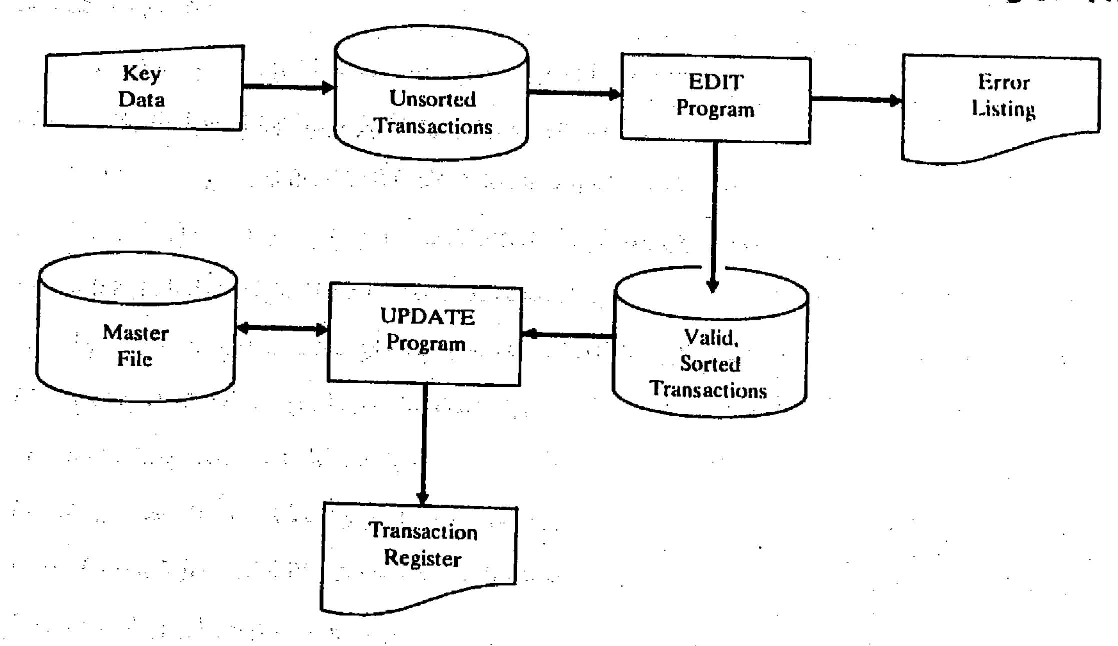
عندما تخزن ملفات رئيسية مرتبة تتابعيا على قرص .. فمن الممكن تجديد السجلات المنطقية ، دون إنتاج ملف رئيسى جديد : ويحدث هذا عن طريق قراءة السجلات فى الذاكرة الرئيسية ، وإجراء التغييرات عليها ، ثم إعادة كتابتها فى نفس مواقعها على القرص بالضبط . انظر إلى خريطة مسار النظام فى شكل (١١ - ٤) . تقع القيود التالية على هذه العملية :

(١) لا تضاف سجلات إلى الملف .

(٢) يجب أن يحدث الحذف منطقيا (بعمل إشارة على السجلات التى لها رمز حذف) .

(٣) يجب ألا يتغير طول السجل المنطقى .

(٤) يجب أن يفتح الملف كملف مدخلات ومخرجات I-O (انظر الفصل السادس) . .




شكل ( ١١ - ٤ )

يستخدم فعل REWRITE فى إخراج سجلات منطقية ، على نفس موقع القرص الذى سبق قراءته منه ، ويأخذ الشكل التالى :

REWRITE record-name [FROM identifier]

مثال ١١ - ٤ :

```
FD   MASTER-FILE
01   MASTER-RECORD...
..........................................
     OPEN   I-O   MASTER-FILE
     READ MASTER-FILE  AT END...
     PERFORM  MAKE-CHANGES-TO-MASTER-RECORD
     REWRITE  MASTER-RECORD
```

لاحظ أنه لكى يجرى تجديداً فى نفس المكان .. فإنه يجب أن يفتح الملف على أنه ملف مدخلات ومخرجات I-O ، ويسمح ذلك بأن تعمل عبارة READ كمدخلات معتادة ، وعبارة REWRITE تستخدم فى إخراج السجل المنطقى فى نفس موقعه الواقعى من القرص (نفس الموقع الذى استخدمته أخر READ فى القراءة) .

## أسئلة مراجعة

١١ - ١ ما (أ) إنتاج ملف . (ب) استرجاع ملف . (جـ) صيانة ( تجديد ) ملف .

١١ - ٢ ناقش العمليات الثلاث التى يجب أن تؤدى كجزء من الصيانة .

١١ - ٣ ما الغرض من منطقة FILE STATUS ؟ كيف تعرف وكيف تستخدم ؟

١١ - ٤ عرف : (أ) الملف الرئيسى (ب) ملف العمليات الجارية (جـ) حقل رئيسى (حقل مفتاح) .

١١ - ٥ اذكر تسلسل الخطوات المتبعة فى تجديد ملف تتابعى تقليدى .

١١ - ٦ وضح اصطلاح الملف الرئيسى القديم ، واصطلاح الملف الرئيسى الجديد .

١١ - ٧ ناقش طريقة الأجداد والآباء والأطفال للاحتياطى .

١١ - ٨ ما المقصود بمسجل العمليات الجارية ؟

١١ - ٩ ارسم خريطة مسار نظام لتجديد ملف تتابعى تقليدى .

١١ - ١٠ وضح خوارزمى خط الاتزان ، مستخدما شفرة شبيهة .

١١ - ١١ ماذا يعنى التجديد فى نفس المكان ؟

١١ - ١٢ قارن فتح الملف فى : (أ) INPUT (ب) OUTPUT (جـ) I-O .

١١ - ١٣ ما القيود التى يجب وضعها على تجديد الملف التتابعى فى نفس المكان ؟

١١ - ١٤ وضح استخدام فعل REWRITE .

١١ - ١٥ ارسم خريطة مسار نظام لتجديد ملف تتابعى فى نفس المكان .

١١ - ١٦ اذكر إحدى مميزات وإحدى عيوب التجديد فى نفس المكان ، بالمقارنة بالتجديد التقليدى (ملف رئيسى قديم ، وملف رئيسى جديد) . (ملاحظة : اعتبر أنه لن يجرى تغيير على السجلات وخذ فى الاعتبار عملية الاحتياطى) .

## مسائل محلولة

١١ - ١٧ ما الفرق بين تجديد الملف وتجديد السجل ؟

يشير تجديد الملف إلى صيانة الملف ، والذى تحدث فيه إضافة سجلات للملف ، أو حذف سجلات من الملف ، أو تغيير فى محتويات سجلات الملف . ويشير تجديد السجل إلى عملية تغيير السجلات المنطقية أثناء تجديد الملف .

١١ - ١٨ حدد الخطأ فى البيانات الاختبارية لتجديد الملف التتابعى التالية :

```
TRANSACTION-FILE:   4, 23, 15, 12, 30, 50
MASTER-FILE:       10, 20, 30, 40, 50, 60, 70
```

يجب أن يكون ملف العمليات الجارية مرتباً بنفس الحقل المستخدم فى ترتيب الملف الرئيسى . ويجب أن تحتوى سجلات العمليات الجارية على الحقل الرئيسى (المفتاح) ، المرتبة على أساسه سجلات الملف الرئيسى المراد تجديده .

١١ - ١٩ بالإشارة الى السؤال الثانى عشر المذكور عاليه ، اذكر بديلاً آخر لعبارة OPEN .

OPEN EXTEND ؛ حيث يمكن إضافة سجلات جديدة الى نهاية الملف ، باستخدام فعل WRITE .

١١ - ٢٠ كم عدد أجيال ملفات العمليات الجارية ، والملفات الرئيسية القديمة التى يجب حفظها ، ولأى فترة زمنية ، عند استخدام احتياطى الأجداد والآباء والأطفال ؟

لا توجد إجابة واحدة : فأهمية التطبيق والبيانات لمجال الأعمال ، والمتطلبات القانونية التى تحددها الحكومة ، ومتطلبات الحسابات وما إلى ذلك ، يجب أن تؤخذ كلها فى الحسبان .

١١ - ٢١ كيف يعالج الاحتياطى عندما تجدد الملفات فى نفس المكان ؟

يمكن إعداد نسخة من الملف قبل إدخال العمليات الجارية . ويمكن حفظ هذه النسخة مع العمليات الجارية ، وتستخدم فى إعادة بناء الملف الرئيسى فى حالة حدوث تلف للملف الرئيسى .

١١ - ٢٢ تشكل المسألتان ٣٠ ، ٣١ من الفصل الثامن ما يشبه تسلسل تجديد ملف تتابعى ، بينما تكون المسألة ٣٠ برنامج تنقيح، وتؤدى المسألة ٣١ تجديد الملف . راجع المسألة ٣٠ لاستخدام جدول بأعداد الأيام فى كل شهر للتأكد من صحة حقل التاريخ فى بطاقات مدخلات الوقت . (اهمل الحالات الخاصة بنهاية القرن ، وبالسنوات الكبيسة) .

يعطى جزء من برنامج كوبل فى شكل (١١ - ٥) ، وتعرف السطور من ٢٣٤ - ٢٥٦ ، جدول التاريخ ، تجرى السطور من ٢٦٩ - ٢٩٠ ، ومن ٢٩٩ - ٣١٠ التأكد من المدخلات .

```
(THE FOLLOWING TABLE IS IN WORKING-STORAGE) . . .

       01  WS-VALID-DATES-TABLE.
           05  VALID-DAYS-PER-MONTH-VALUES.
               10  FILLER              PIC 99          VALUE 31.
               10  FILLER              PIC 99          VALUE 28.
               10  FILLER              PIC 99          VALUE 31.
               10  FILLER              PIC 99          VALUE 30.
               10  FILLER              PIC 99          VALUE 31.
               10  FILLER              PIC 99          VALUE 30.
               10  FILLER              PIC 99          VALUE 31.
               10  FILLER              PIC 99          VALUE 31.
               10  FILLER              PIC 99          VALUE 30.
               10  FILLER              PIC 99          VALUE 31.
               10  FILLER              PIC 99          VALUE 30.
               10  FILLER              PIC 99          VALUE 31.
           05  DAYS-PER-MONTH      REDEFINES VALID-DAYS-PER-MONTH-VALUES
                                       PIC 99
                                       OCCURS 12 TIMES.
.
.
.
(THE FOLLOWING ARE THE RELEVANT PROCEDURE DIVISION PARAGRAPHS) . . .

          240-VALIDATE-FIELDS.

              MOVE "NO"    TO WS-ERROR-DETECTED-SW
              MOVE SPACES TO WS-UNDER-LINE
              IF TIME-CARD-ID NOT NUMERIC
                  MOVE ALL "-" TO WS-UNDER-ID
                  MOVE "YES"   TO WS-ERROR-DETECTED-SW
              .
              IF TIME-CARD-HOURS NOT NUMERIC
                  MOVE ALL "-" TO WS-UNDER-HOURS
                  MOVE "YES"   TO WS-ERROR-DETECTED-SW
              .
              IF TIME-CARD-CLOSING-DATE NOT NUMERIC
                  MOVE ALL "-" TO WS-UNDER-DATE
                  MOVE "YES"   TO WS-ERROR-DETECTED-SW
              ELSE
                  PERFORM 260-VALIDATE-DATE
              .
              IF TIME-CARD-DEPARTMENT NOT NUMERIC
                  MOVE ALL "-" TO WS-UNDER-DEPARTMENT
                  MOVE "YES"   TO WS-ERROR-DETECTED-SW
              .

          260-VALIDATE-DATE.

              IF TIME-CARD-CLOSING-YR GREATER THAN SYSTEM-YY
                  MOVE ALL "-" TO WS-UNDER-YR
                  MOVE "YES" TO WS-ERROR-DETECTED-SW
              .
              IF TIME-CARD-CLOSING-MO LESS THAN 1 OR GREATER THAN 12
                  MOVE ALL "-" TO WS-UNDER-MO
                  MOVE "YES" TO WS-ERROR-DETECTED-SW
              ELSE IF TIME-CARD-CLOSING-DAY LESS THAN 1 OR
                  GREATER THAN DAYS-PER-MONTH (TIME-CARD-CLOSING-MO)
                      MOVE ALL "-" TO WS-UNDER-DAY
                      MOVE "YES" TO WS-ERROR-DETECTED-SW
```

شكل ( ١١ - ٥ )

١١ – ٢٣ اكتب شفرة برنامج كوبل كاملا للمسألة رقم ٣١ من الفصل الثامن . واستخدم فحص جدول الضرائب فى حساب ضريبة الدخل ، وجدد الملف الرئيسى للعاملين (فى موقعه) ، وذلك بإضافة إجمالى الأجر ، وقيمة الضرائب ، وصافى الأجر الى الحقول المناظرة لها من أول العام حتى الآن داخل السجل الرئيسى للعاملين .

انظر شكل ١١ – ٦ . توجد حسابات الضرائب فى السطور من ١١٢ – ١٣٧ ، ومن ٣٣١ – ٣٤٣ التجديد فى السطور من ٢٣٤ – ٢٥٦ ، ومن السطور ٣٠٩ – ٣١٥ .

```
00001          IDENTIFICATION DIVISION.
00002
00003          PROGRAM-ID.   PAYROLL.
00004
00005          AUTHOR.   LARRY NEWCOMER.
00006          INSTALLATION.   PENN STATE UNIVERSITY -- YORK CAMPUS.
00007
00008          ENVIRONMENT DIVISION.
00009
00010          CONFIGURATION SECTION.
00011          SOURCE-COMPUTER.   IBM-3081.
00012          OBJECT-COMPUTER.   IBM-3081.
00013
00014          INPUT-OUTPUT SECTION.
00015          FILE-CONTROL.
00016
00017              SELECT VALID-TIME-FILE
00018                  ASSIGN TO DISKTIME
00019                  ORGANIZATION IS SEQUENTIAL
00020                  ACCESS IS SEQUENTIAL
00021              .
00022              SELECT EMPLOYEE-MASTER-FILE
00023                  ASSIGN TO EMPMAST
00024                  ORGANIZATION IS SEQUENTIAL
00025                  ACCESS IS SEQUENTIAL
00026              .
00027              SELECT PAYROLL-CHECK-FILE
00028                  ASSIGN TO PAYCHECK
00029                  ORGANIZATION IS SEQUENTIAL
00030                  ACCESS IS SEQUENTIAL
00031              .
00032              SELECT PAYROLL-REGISTER-FILE
00033                  ASSIGN TO REGISTER
00034                  ORGANIZATION IS SEQUENTIAL
00035                  ACCESS IS SEQUENTIAL
00036              .
00037
00038          DATA DIVISION.
00039
00040          FILE SECTION.
00041
00042          FD  VALID-TIME-FILE
00043              BLOCK CONTAINS 0 RECORDS
00044              RECORD CONTAINS 80 CHARACTERS
00045              LABEL RECORDS ARE STANDARD
00046              .
00047
00048          01  VALID-TIME-RECORD.
00049              05  VALID-TIME-ID                 PIC X(5).
00050              05  VALID-TIME-HOURS              PIC S9(2)V9.
00051              05  VALID-TIME-CLOSING-DATE       PIC X(6).
00052              05  VALID-TIME-DEPARTMENT         PIC X(4).
00053              05  VALID-TIME-EDIT-DATE          PIC X(6).
```

شكل ( ١١ – ٦ )

```
            05  FILLER                         PIC X(56).

       FD  EMPLOYEE-MASTER-FILE
           BLOCK CONTAINS 0 RECORDS
           RECORD CONTAINS 80 CHARACTERS
           LABEL RECORDS ARE STANDARD
           .

       01  EMPLOYEE-MASTER-RECORD.
           05  EMPLOYEE-MASTER-ID         PIC X(5).
           05  EMPLOYEE-MASTER-NAME       PIC X(20).
           05  EMPLOYEE-MASTER-RATE       PIC S9(3)V99.
           05  EMPLOYEE-MASTER-YTD-GROSS
                                          PIC 9(7)V99.
           05  EMPLOYEE-MASTER-YTD-TAX    PIC 9(6)V99.
           05  EMPLOYEE-MASTER-YTD-NET    PIC 9(7)V99.
           05  FILLER                     PIC X(24).

       FD  PAYROLL-CHECK-FILE
           BLOCK CONTAINS 0 RECORDS
           RECORD CONTAINS 66 CHARACTERS
           LABEL RECORDS ARE STANDARD
           .

       01  PAYROLL-CHECK-RECORD.
           05  PAYROLL-ID                 PIC X(5).
           05  PAYROLL-HOURS              PIC S9(2)V9.
           05  PAYROLL-DATE               PIC X(6).
           05  PAYROLL-DEPARTMENT         PIC X(4).
           05  PAYROLL-RATE               PIC S9(3)V99.
           05  PAYROLL-NAME               PIC X(20).
           05  PAYROLL-GROSS              PIC S9(6)V99.
           05  PAYROLL-TAX                PIC S9(5)V99.
           05  PAYROLL-NET                PIC S9(6)V99.

       FD  PAYROLL-REGISTER-FILE
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .

       01  ERROR-REPORT-LINE              PIC X(132).

       WORKING-STORAGE SECTION.

    01  PROGRAM-SWITCHES.
        05  WS-END-TIME-FILE-SW        PIC X(3).
            88  NO-MORE-TIMES          VALUE "YES".
            88  TIME-AVAILABLE         VALUE "NO".
        05  WS-END-MASTER-FILE-SW      PIC X(3).
            88  NO-MORE-MASTERS        VALUE "YES".
            88  MASTER-AVAILABLE       VALUE "NO ".

    01  PROGRAM-COUNTERS.
        05  WS-PAGE-NUMBER             PIC S9(3)      COMP-3.
        05  WS-NUMBER-EMPLOYEES        PIC S9(5)      COMP-3.
        05  WS-NUMBER-TO-SKIP          PIC S9         COMP SYNC.
        05  WS-LINES-ON-PAGE           PIC S9(2)      COMP SYNC.
```

شكل ( ١١ - ٦ ) تكملة

```
      *    ------------------------------------------------------------
      *
      *    TAX TABLE IS USED TO LOOKUP TAX-RATE.  NOTE HOW ENTRIES
      *    ARE ARRANGED IN TABLE.  LAST ENTRY = ZERO GUARANTEES
      *    SEARCH WILL ALWAYS SUCCEED.  ALTHOUGH WE USE THE VALUE
      *    AND REDEFINES CLAUSES TO LOAD THIS TABLE.
      *    IT COULD ALSO BE INPUT FROM A FILE.
      *
      *    ------------------------------------------------------------

       01  TAX-DATA.
           05  TAX-TABLE-VALUES.
               10  FILLER             PIC S9(3)V99 COMP-3 VALUE +800.00.
               10  FILLER             PIC SV999    COMP-3 VALUE +.25.
               10  FILLER             PIC S9(3)V99 COMP-3 VALUE +500.00.
               10  FILLER             PIC SV999    COMP-3 VALUE +.15.
               10  FILLER             PIC S9(3)V99 COMP-3 VALUE +300.00.
               10  FILLER             PIC SV999    COMP-3 VALUE +.08.
               10  FILLER             PIC S9(3)V99 COMP-3 VALUE ZERO.
               10  FILLER             PIC SV999    COMP-3 VALUE +.05.
           05  TAX-TABLE              REDEFINES TAX-TABLE-VALUES
                                      OCCURS 4 TIMES
                                      INDEXED BY TAX-INDEX.
               10  LOWER-LIMIT        PIC S9(3)V99 COMP-3.
               10  TAX-RATE           PIC SV999    COMP-3.

       Ø1  PROGRAM-TOTAL-AREAS.
           05  WS-TOTAL-HOURS          PIC S9(7)V9     COMP-3.

       01  PROGRAM-COMPUTATION-AREAS.
           05  WS-GROSS-PAY            PIC S9(7)V99    COMP-3.
           05  WS-NET-PAY              PIC S9(7)V99    COMP-3.
           05  WS-TAX                  PIC S9(5)V99    COMP-3.

       01  PROGRAM-CONSTANTS.
           05  WS-PAGE-SIZE            PIC S9(2)       COMP SYNC
                                       VALUE +50.
           05  WS-SKIP-BEFORE-HEADING  PIC S9(2)       COMP SYNC
                                       VALUE +2.
           05  WS-SKIP-BEFORE-FOOTING  PIC S9(2)       COMP SYNC
                                       VALUE +4.
           05  WS-SKIP-BEFORE-DETAIL   PIC S9(2)       COMP SYNC
                                       VALUE +2.
           05  WS-UNMATCHED-TIME-MESSAGE
                                       PIC X(30)
                                       VALUE "EMPLOYEE NOT ON FILE".
           05  WS-UNMATCHED-MASTER-MESSAGE
                                       PIC X(30)
                                       VALUE "INACTIVE THIS PAY PERIOD".
           05  WS-OVERTIME-POINT       PIC S99V9       COMP-3
                                       VALUE +40.0.
           05  WS-OVERTIME-FACTOR      PIC S9V9        COMP-3
                                       VALUE +1.5.

       01  WS-SYSTEM-DATE.
           05  SYSTEM-YY                   PIC 99.
           05  SYSTEM-MM                   PIC 99.
           05  SYSTEM-DD                   PIC 99.
```

شكل ( ١١ - ٦ ) تكملة

```
       01  WS-HEADING-LINE-1.
           05  FILLER                  PIC X(2)      VALUE SPACES.
           05  FILLER                  PIC X(18)
                                       VALUE "PAYROLL REGISTER".
           05  WS-HEADING-MM           PIC Z9.
           05  FILLER                  PIC X         VALUE "/".
           05  WS-HEADING-DD           PIC 99.
           05  FILLER                  PIC X         VALUE "/".
           05  WS-HEADING-YY           PIC 99.
           05  FILLER                  PIC X(3)      VALUE SPACES.
           05  FILLER                  PIC X(5)      VALUE "PAGE ".
           05  WS-HEADING-PAGE         PIC ZZ9.
           05  FILLER                  PIC X(93)     VALUE SPACES.

       01  WS-HEADING-LINE-2.
           05  FILLER                  PIC X(7)      VALUE "  ID".
           05  FILLER                  PIC X(5)      VALUE "HRS".
           05  FILLER                  PIC X(8)      VALUE " DATE".
           05  FILLER                  PIC X(6)      VALUE "DEPT".
           05  FILLER                  PIC X(8)      VALUE " RATE".
           05  FILLER                  PIC X(98)     VALUE "NET PAY".

       01  WS-DETAIL-LINE.
           05  WS-DETAIL-ID            PIC X(5).
           05  FILLER                  PIC X(1)      VALUE SPACES.
           05  WS-DETAIL-HOURS         PIC ZZ.9.
           05  FILLER                  PIC X(2)      VALUE SPACES.
           05  WS-DETAIL-DATE          PIC X(6).
           05  FILLER                  PIC X(2)      VALUE SPACES.
           05  WS-DETAIL-DEPARTMENT    PIC X(4).
           05  FILLER                  PIC X(2)      VALUE SPACES.
           05  WS-DETAIL-RATE          PIC ZZZ.99.
           05  FILLER                  PIC X(2)      VALUE SPACES.
           05  WS-DETAIL-REMARKS.
               10  WS-DETAIL-PAY       PIC ZZZ,ZZZ.99.
               10  FILLER              PIC X(88).

       01  WS-FOOTING-LINE.
           05  FILLER                  PIC X(12)
                                       VALUE "# EMPLOYEES=".
           05  WS-FOOTING-COUNT        PIC ZZ,ZZ9.
           05  FILLER                  PIC X(3)      VALUE SPACES.
           05  FILLER                  PIC X(12)
                                       VALUE "TOTAL HOURS=".
           05  WS-FOOTING-TOTAL        PIC Z,ZZZ,ZZZ.9.
           05  FILLER                  PIC X(88)     VALUE SPACES.

       01  WS-OUTPUT-AREA              PIC X(132).

       PROCEDURE DIVISION.

       CREATE-PAYROLL-CHECK-FILE.

           PERFORM 100-INITIALIZE
           PERFORM 200-MATCH-TIME-CARDS-TO-MASTER
               UNTIL NO-MORE-TIMES AND NO-MORE-MASTERS
           PERFORM 300-TERMINATE
           STOP RUN
```

شكل ( ١١ - ٦ ) تكملة

```
       100-INITIALIZE.
           OPEN    INPUT    VALID-TIME-FILE
                   I-O      EMPLOYEE-MASTER-FILE
                   OUTPUT   PAYROLL-REGISTER-FILE
                            PAYROLL-CHECK-FILE
           MOVE "NO" TO                 WS-END-TIME-FILE-SW
                                        WS-END-MASTER-FILE-SW
           MOVE ZERO TO                 WS-PAGE-NUMBER
                                        WS-NUMBER-EMPLOYEES
                                        WS-TOTAL-HOURS
           MOVE WS-PAGE-SIZE TO         WS-LINES-ON-PAGE
           ACCEPT WS-SYSTEM-DATE FROM DATE
           MOVE SYSTEM-YY TO            WS-HEADING-YY
           MOVE SYSTEM-MM TO            WS-HEADING-MM
           MOVE SYSTEM-DD TO            WS-HEADING-DD
           PERFORM 270-GET-TIME-RECORD
           PERFORM 280-GET-MASTER-RECORD
           .
       B-MATCH-TIME-CARDS-TO-MASTER.
           MOVE SPACES TO WS-DETAIL-LINE
           IF VALID-TIME-ID EQUAL EMPLOYEE-MASTER-ID
               PERFORM 210-CREATE-PAYROLL-REC
               PERFORM 215-UPDATE-MASTER-FILE
               PERFORM 270-GET-TIME-RECORD
               PERFORM 280-GET-MASTER-RECORD
           ELSE IF VALID-TIME-ID LESS THAN EMPLOYEE-MASTER-ID
               MOVE WS-UNMATCHED-TIME-MESSAGE
                                        TO WS-DETAIL-REMARKS
               MOVE VALID-TIME-ID       TO WS-DETAIL-ID
               MOVE VALID-TIME-HOURS    TO WS-DETAIL-HOURS
               MOVE VALID-TIME-CLOSING-DATE
                                        TO WS-DETAIL-DATE
               MOVE VALID-TIME-DEPARTMENT
                                        TO WS-DETAIL-DEPARTMENT
               MOVE WS-SKIP-BEFORE-DETAIL
                                        TO WS-NUMBER-TO-SKIP
               MOVE WS-DETAIL-LINE      TO WS-OUTPUT-AREA
               PERFORM 270-GET-TIME-RECORD
           ELSE
               MOVE WS-UNMATCHED-MASTER-MESSAGE
                                        TO WS-DETAIL-REMARKS
               MOVE EMPLOYEE-MASTER-ID     TO WS-DETAIL-ID
               MOVE EMPLOYEE-MASTER-RATE   TO WS-DETAIL-RATE
               MOVE WS-SKIP-BEFORE-DETAIL  TO WS-NUMBER-TO-SKIP
               MOVE WS-DETAIL-LINE         TO WS-OUTPUT-AREA
               PERFORM 280-GET-MASTER-RECORD
           .
           PERFORM 260-PRINT-REGISTER
           .
       210-CREATE-PAYROLL-REC.
           MOVE VALID-TIME-ID              TO WS-DETAIL-ID
           MOVE VALID-TIME-HOURS           TO WS-DETAIL-HOURS
           MOVE VALID-TIME-CLOSING-DATE    TO WS-DETAIL-DATE
           MOVE VALID-TIME-DEPARTMENT      TO WS-DETAIL-DEPARTMENT
           MOVE EMPLOYEE-MASTER-RATE       TO WS-DETAIL-RATE
```

شكل ( ١١ - ٦ ) تكملة

```
           MOVE WS-SKIP-BEFORE-DETAIL     TO WS-NUMBER-TO-SKIP

           PERFORM 220-CALCULATE-PAY
           MOVE WS-NET-PAY                TO WS-DETAIL-PAY
           PERFORM 250-WRITE-PAYROLL-RECORD
           MOVE WS-DETAIL-LINE TO WS-OUTPUT-AREA
         ADD 1                          TO WS-NUMBER-EMPLOYEES
         ADD VALID-TIME-HOURS           TO WS-TOTAL-HOURS
     . .

     220-CALCULATE-PAY.

         PERFORM 230-CALCULATE-GROSS
         PERFORM 240-CALCULATE-TAX
         SUBTRACT WS-TAX FROM WS-GROSS-PAY
             GIVING WS-NET-PAY
         .

     215-UPDATE-MASTER-FILE.

         ADD WS-TAX       TO EMPLOYEE-MASTER-YTD-TAX
         ADD WS-GROSS-PAY TO EMPLOYEE-MASTER-YTD-GROSS
         ADD WS-NET-PAY   TO EMPLOYEE-MASTER-YTD-NET
         REWRITE EMPLOYEE-MASTER-RECORD
         .

     230-CALCULATE-GROSS.

         IF VALID-TIME-HOURS GREATER THAN WS-OVERTIME-POINT
             COMPUTE WS-GROSS-PAY =
                     (WS-OVERTIME-POINT * EMPLOYEE-MASTER-RATE)
                 +   ((VALID-TIME-HOURS - WS-OVERTIME-POINT)
                         * EMPLOYEE-MASTER-RATE * WS-OVERTIME-FACTOR)
         ELSE
             COMPUTE WS-GROSS-PAY =
                 VALID-TIME-HOURS * EMPLOYEE-MASTER-RATE


      240-CALCULATE-TAX

     *    --------------------------------------------------------
     *
     *    USE SEQUENTIAL SEARCH TO LOOKUP TAX RATE IN TAX TABLE
     *    (DUE TO DESIGN OF TABLE, AT END... CONDITION CAN'T OCCUR)
     *
     *    --------------------------------------------------------

          SET TAX-INDEX TO 1
          SEARCH TAX-TABLE
              WHEN WS-GROSS-PAY GREATER THAN LOWER-LIMIT (TAX-INDEX)
                  COMPUTE WS-TAX =
                      WS-GROSS-PAY * TAX-RATE (TAX-INDEX)


      250-WRITE-PAYROLL-RECORD.

          MOVE VALID-TIME-ID             TO PAYROLL-ID
          MOVE VALID-TIME-HOURS          TO PAYROLL-HOURS
          MOVE VALID-TIME-CLOSING-DATE   TO PAYROLL-DATE
          MOVE VALID-TIME-DEPARTMENT     TO PAYROLL-DEPARTMENT
```

شكل ( ١١ - ٦ ) تكملة

```
           MOVE EMPLOYEE-MASTER-RATE      TO PAYROLL-RATE
           MOVE EMPLOYEE-MASTER-NAME      TO PAYROLL-NAME
           MOVE WS-GROSS-PAY              TO PAYROLL-GROSS
           MOVE WS-TAX                    TO PAYROLL-TAX
           MOVE WS-NET-PAY                TO PAYROLL-NET
           WRITE PAYROLL-CHECK-RECORD
           .

       260-PRINT-REGISTER.

           IF WS-LINES-ON-PAGE + WS-NUMBER-TO-SKIP
               GREATER THAN WS-PAGE-SIZE
                   ADD 1 TO WS-PAGE-NUMBER
                   MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE
                   WRITE ERROR-REPORT-LINE
                       FROM WS-HEADING-LINE-1
                       AFTER ADVANCING PAGE
                   WRITE ERROR-REPORT-LINE
                       FROM WS-HEADING-LINE-2
                       AFTER ADVANCING WS-SKIP-BEFORE-HEADING LINES
                   MOVE WS-SKIP-BEFORE-HEADING TO WS-LINES-ON-PAGE
           .
           WRITE ERROR-REPORT-LINE
               FROM WS-OUTPUT-AREA
               AFTER ADVANCING WS-NUMBER-TO-SKIP LINES
           ADD WS-NUMBER-TO-SKIP TO WS-LINES-ON-PAGE
           .

       270-GET-TIME-RECORD.

           READ VALID-TIME-FILE
               AT END
                   MOVE "YES" TO WS-END-TIME-FILE-SW
                   MOVE HIGH-VALUES TO VALID-TIME-ID


       280-GET-MASTER-RECORD.

           READ EMPLOYEE-MASTER-FILE
               AT END
                   MOVE "YES" TO WS-END-MASTER-FILE-SW
                   MOVE HIGH-VALUES TO EMPLOYEE-MASTER-ID
           .

       300-TERMINATE.

           MOVE WS-NUMBER-EMPLOYEES     TO WS-FOOTING-COUNT
           MOVE WS-TOTAL-HOURS          TO WS-FOOTING-TOTAL
           MOVE WS-FOOTING-LINE         TO WS-OUTPUT-AREA
           MOVE WS-SKIP-BEFORE-FOOTING  TO WS-NUMBER-TO-SKIP
           MOVE WS-FOOTING-LINE         TO WS-OUTPUT-AREA
           PERFORM 260-PRINT-REGISTER

           CLOSE   VALID-TIME-FILE
                   EMPLOYEE-MASTER-FILE
                   PAYROLL-CHECK-FILE
                   PAYROLL-REGISTER-FILE

```

شكل ( ١١ - ٦ ) تكملة

# الفصل الثانى عشر

# ترتيب ودمج الملفات

# File Sorting and Merging

يشمل ترتيب ودمج الملفات إعادة ترتيب السجلات المنطقية فى الملف ، والتى تكون موجودة فى وحدة تخزين مساعد . وهذا يضاهى ترتيب الجدول ، الذى يحدث فيه إعادة ترتيب لأجزاء من سجل منطقى ( أى الجدول ) . ويعد ترتيب الملفات سهلاً فى برمجته فى الكوبل باستخدام فعل الترتيب SORT .

## ١٢ - ١ معجم ترتيب الملفات

تسمى الحقول الموجودة فى سجل منطقى ، والتى يرتب على أساسها الملف ، حقول تحكم control fields ، أو مفاتيح الترتيب sort keys . يمكن ترتيب السجلات فى ملف فى تتابع تصاعدى ascending ، أو تتابع تنازلى descending طبقا لأى حقل . يسمح كوبل IBM OS/VS بالترتيب الآتى الملف ، طبقا لعدد يصل إلى 12 حقل مختلف ، وبأى خليط من التتابع التصاعدى أو التنازلى . عندما يؤدى الترتيب على مفاتيح متعددة .. يسمى الحقل الأكثر أهمية بالمفتاح الرئيسى major key ويسمى الحقل الأقل أهمية بالمفتاح الأدنى minor key ، أما بقية الحقول - فتسمى بالمفاتيح المتوسطة intermediate keys - والمفتاح الادنى كذلك - فهى تؤثر على الترتيب داخل كل قيمة من قيم المفتاح الرئيسى . وإذا ركزنا بالتالى الانتباه على حقل متوسط أو حقل أدنى فقط .. فلن يبدو الملف ككل بأنه مرتب .

**مثال ١٢ - ١ :**

يعرض جدول ( ١٢ - ١ ) ترتيباً لملف مبيعات طبقا لثلاثة مفاتيح . الترتيب الرئيسى تصاعدى ، والترتيب المتوسط - داخل الترتيب الرئيسى - تنازلى ، والترتيب الأدنى - داخل الترتيب المتوسط - تصاعدى .

جدول ( ١٢ - ١ )

| قيمة المبيعات ( ليس حقل مفتاح ) | رقم العميل ( مفتاح أدنى ) | رقم البائع ( مفتاح متوسط ) | رقم القسم ( مفتاح رئيسى ) |
|---|---|---|---|
| 1,000.98 | 10925 | 7823 | 101 |
| 2,000.00 | 27832 | 7823 | 101 |
| 1,500.00 | 59838 | 7823 | 101 |
| 1,200.32 | 20782 | 6345 | 101 |
| 5,000.00 | 48793 | 6345 | 101 |
| 3,003.43 | 10925 | 3287 | 101 |
| 1,305.50 | 27832 | 8291 | 206 |
| 2,783.45 | 40058 | 8291 | 206 |
| 5,672.34 | 39840 | 5094 | 206 |
| 3,476.52 | 60291 | 5094 | 206 |
| 4,000.00 | 27832 | 4925 | 206 |
| 2,987.50 | 30618 | 4925 | 206 |
| 4,526.98 | 10071 | 3920 | 206 |

## ١٢ - ٢ استخدام عبارة الترتيب

عبارة الترتيب – الموضح تكوينها فى شكل (١٢ – ١) تكتب فى جزء الإجراءات ، ومعها يجب استخدام ملف ترتيب sort file افتراضى ، ويجب أن تكون لهذا الملف عبارة SELECT فى جزء الأوساط ، وجزء وصف ملف SD يشبه وصف الملف FD فى جزء البيانات . ويعرف وصف السجل لملف الترتيب الحقول الرئيسية (حقول المفاتيح ) التى يؤدى الترتيب طبقا لها . وتستخدم منطقة السجل المنطقى لملف الترتيب فى تمرير سجلات من وإلى برنامج الكوبل ، وبرنامج منفعة سابق الإعداد للترتيب والدمج sort merge utility program ، والذى يجرى الترتيب الفعلى . وتأتى برامج المنفعة مع نظام التشغيل .

```
SORT file-name-1
    ON {ASCENDING | DESCENDING} KEY data-name-1 [data-name-2] . . .
    [ON {ASCENDING | DESCENDING} KEY data-name-3 [data-name-4] . . .] . . .
    [COLLATING SEQUENCE IS alphabet-name]
    {USING file-name-2 [file-name-3] . . .
    |INPUT PROCEDURE IS section-name-1 [{THROUGH | THRU} section-name-2]}
    {GIVING file-name-4
    |OUTPUT PROCEDURE IS section-name-3 [{THROUGH | THRU} section-name-4]}
```

شكل ( ١٢ - ١ )

مثال ١٢ - ٢ :

```
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
ENVIRONMENT DIVISION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT MORTGAGE-PAYMENT-FILE
        ASSIGN TO MORTRECV
        ORGANIZATION IS SEQUENTIAL
        ACCESS IS SEQUENTIAL
        .
    SELECT PAYMENT-SORT-FILE
        ASSIGN TO SORTWK
        .
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
DATA DIVISION.
FILE SECTION.
FD  MORTGAGE-PAYMENT-FILE
    BLOCK CONTAINS 0 CHARACTERS
    RECORD CONTAINS 23 CHARACTERS
    LABEL RECORDS ARE STANDARD

01  MORTGAGE-PAYMENT-RECORD.
    05 MORTGAGE-ACCOUNT-ID          PIC X(6).
    05 MORTGAGE-DUE-DATE            PIC X(6).
    05 MORTGAGE-PAID-DATE           PIC X(6).
    05 MORTGAGE-AMOUNT              PIC S9(3)V99.
SD  PAYMENTS-SORT-FILE
    RECORD CONTAINS 23 CHARACTERS

01  SORT-RECORD.
    05 SORT-ACCOUNT-ID              PIC X(6).
    05 SORT-DUE-DATE                PIC X(6).
    05 SORT-PAID-DATE               PIC X(6).
    05 SORT-AMOUNT                  PIC S9(3)V99.
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
PROCEDURE DIVISION.
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    SORT PAYMENTS-SORT-FILE
        ON ASCENDING KEY SORT-ACCOUNT-ID
        ON ASCENDING KEY SORT-PAID-DATE
        INPUT PROCEDURE IS SCREEN-PAYMENT-RECORDS
        OUTPUT PROCEDURE IS PRODUCE-QUARTERLY-REPORT
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
SCREEN-PAYMENT-RECORDS SECTION.
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
PRODUCE-QUARTERLY-REPORT SECTION.
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

هنا .. يعتبر الملف MORTAGE - PAYMENT - FILE هو الملف الفعلى على القرص ، والمراد ترتيبه تصاعديا طبقا لتاريخ الدفع (المفتاح الأدنى) ، داخل رقم تعريف الحساب (المفتاح الرئيسى) ، وملف الترتيب هو PAYMENTS - SORT - FILE . لاحظ ما يلى :

(١) بالنسبة لملف الترتيب .. تأخذ عبارة SELECT الشكل المبسط التالى :

```
SELECT sort-file-name
    ASSIGN TO assignment-name-comment
```

ونظراً لأن ملف الترتيب ليس ملفا فعليا .. يعامل اسم ASSIGN TO كتعليق . ولا تكون هناك حاجة إلى لغة تحكم العمل إضافية ؛ لتعريف ملف الترتيب .

(٢) عادة ما تكون عبارات تحكم العمل أو عبارات لغة الأوامر مطلوبة إذا كانت عبارة SORT تستدعى منفعة الترتيب أو الدمج . وعلى القارىء أن يحدد المطلوب بالنسبة للجهاز المتاح له استخدامه .

(٣) الملف PAYMENT - SORT - FILE معرف فى جزء البيانات بواسطة SD :

```
SD   sort-file-name
     [RECORD CONTAINS [integer-1 TO] integer-2 CHARACTERS]
     [DATA {RECORD IS  } data-name]
           {RECORDS ARE}
```

(٤) يقدم SORT - RECORD ( المستوى 01 ) وصفا للسجلات المراد ترتيبها ، ويعمل كذاكرة احتياطيه ، تمرر فيها السجلات من والى منفعة الترتيب الى ومن البرنامج .

(٥) تسمى عبارة SORT ملف الترتيب ، وليس الملف الفعلى . ويعرف جزء ASCENDING KEY أو DESCENDING KEY حقول مفاتيح الترتيب من الحقل الرئيسى إلى الحقل الأدنى ، كما يجب أن تظهر عناصر البيانات هذه فى وصف سجل ملف الترتيب ، الذى يبدأ بتنفيذ INPUT PROCEDURE .

(٦) يجب أن يكون INPUT PROCEDURE قسما . ويحتوى SCREEN - PAYMENT - RECORDS على دورة برنامج تقرأ كل سجل منطقى من MORTAGE - PAYMENT - FILE ، وتنقله إلى SORT - RECORD ، وتسلمه إلى منفعة الترتيب بتنفيذ عبارة RELEASE . وعندما تنفذ عبارة SORT .. فإن INPUT PROCEDURE ينفذ مرة واحدة .

(٧) عندما ينتهى تنفيذ INPUT PROCEDURE .. ترتب منفعة الترتيب – بصورة تلقائية – السجلات المنطقية التى أزيحت إليه . وعند إتمام الترتيب ينفذ OUTPUT PROCEDURE (PRODUCE - QUARTERLY - REPORT) مرة واحدة .

(٨) يجب أن يكون OUTPUT PROCEDURE قسما كذلك ، ويكتب ليطلب إعادة منفعة الترتيب (إلى SORT - RECORD ) السجل المنطقى* التالى فى تتابع الترتيب . ويحدث هذا بتنفيذ عبارة RETURN داخل OUTPUT PROCEDURE. عند ذلك يستطيع .. OUTPUT PROCEDURE تشغيل السجل (وفى هذه الحالة .. فهو يطبع سطرا من سطور التقرير) .

كما فى PRODUCE - QUARTERLY - REPORT .. فإن INPUT PROCEDURE يحتوى على دورة؛ حيث يعاد RETURN الملف المرتب من منفعة الترتيب ، سجلا سجلا .

(٩) عندما ينتهى تنفيذ OUTPUT PROCEDURE .. يستمر تنفيذ البرنامج بالعبارة التى تلى عبارة SORT مباشرة. حيث إن INPUT PROCEDURE يزيح RELEASE سجلات منطقية إلى منفعة الترتيب سجلا سجلا ؛ فيمكن استخدامه فى غربلة screen السجلات ، أو عمل تشغيل مسبق preprocess قبل ترتيبها . وفى الواقع ، إذا لم تكن هناك حاجة الى الغربلة أو التشغيل المسبق للسجلات المراد ترتيبها .. يمكن حذف INPUT PROCEDURE ، وتستطيع منفعة الترتيب الحصول على السجلات المراد ترتيبها من الملف الفعلى مباشرة ، ويحدد هذا البديل بجزء USING فى عبارة SORT .

**مثال ١٢ - ٣ :**

افرض أنه مطلوب طباعة عناوين بريدية من الملف الرئيسى للعملاء ؛ مرتبة طبقا لرقم العميل ، وأنه من المرغوب فيه إرسال خطابات إلى العملاء الذين لم يقوموا بعمليات شراء خلال آخر ٨ شهور فقط . ولكى تقل تكاليف البريد .. يجب أن ترتب العناوين طبقا للرقم البريدى ؛ حيث يفترض أن معظم العملاء كانوا نشطين خلال آخر ٨ شهور ، ؛ لذا فمن الممكن استخدام INPUT PROCEDURE لإزاحة RELEASE السجلات الخاصة بالعملاء ، المراد طباعة عناوينهم فقط إلى منفعة الترتيب .

أكثر من هذا .. يمكن تعريف سجلات ملف الترتيب بأنها تحتوى على حقول الاسم والعنوان فقط من سجلات الملف الرئيسى للعملاء ، ويمكن كتابة INPUT PROCEDURE ؛ لنقل هذه الحقول - فقط - إلى منطقة سجلات ملف الترتيب ، قبل إزاحتها RELEASE إلى منفعة الترتيب .

يمكن للترتيب أن يأخذ بعد ذلك وقتا أقل من الكمبيوتر ، حيث يجرى الترتيب على سجلات أصغر .

**مثال ١٢ - ٤ :**

افرض ان الملف الرئيسى للعملاء يحتوى على إجمالى مشتروات آخر سنة ، وإجمالى مشتروات السنة الحالية . ومن المرغوب فيه طباعة قائمة بالعملاء فى ترتيب تنازلى طبقا لنسبة التغيير المئوية من العام الماضى إلى العام الحالى. لعمل ذلك.. ، يجب وصف حقل خاص بنسبة التغيير المئوية ، كجزء من وصف سجل ملف الترتيب . ويمكن استخدام INPUT PROCEDURE فى إدخال سجلات الملف الرئيسى للعملاء ؛ لحساب نسب التغيير المئوية ، ثم تنقل بعد ذلك هذه النسبة مع المعلومات المطلوبة من سجل الملف الرئيسى للعملاء إلى سجل الترتيب ؛ لإزاحتها RELEASE إلى منطقة الترتيب .

نركز على انه عند استخدام INPUT PROCEDURE ، لا تحتاج السجلات المراد ترتيبها أن تناظر السجلات الموجودة فى أى ملف فعلى (مثل : لا يوجد حقل بنسبة التغيير فى سجلات الملف الرئيسى للعملاء ) . ومن ناحية أخرى .. فإنه عند استخدام USING .. يجب أن يعكس SD كلاً من : USAGE , PICTURE ، وهياكل السجلات فى ملف USING بالضبط

وكما أن INPUT PROCEDURE تمكن من إجراء تشغيل سابق للسجلات غير المرتبة فإن OUTPUT PROCEDURE يجرى تشغيلاً لاحقاً على السجلات المرتبة ؛ فإذا لم تكن هناك حاجة للتشغيل اللاحق .. فمن الممكن استخدام جزء GIVING فى توجيه منفعة الترتيب ، إلى وضع السجلات المرتبة فى ملف GIVING مباشرة .

**مثال ١٢ - ٥ :**

إذا استخدم الملف MORTGAGE - PAYMENT - FILE من مثال (١٢ - ٢) فى ملف تجديد الملف الرئيسى للقروض .. فيمكن أن يحتوى برنامج التنقيح الذى يختبر الصحة ويرتبها على مايلى :

```
SORT PAYMENTS-SORT-FILE
    ON ASCENDING KEY SORT-ACCOUNT-ID
    INPUT PROCEDURE IS VALIDATE-PAYMENT-RECORDS
    GIVING VALID-TRANS-FILE
```

حيث VALID - TRANS - FILE هو ملف فعلى ، له عبارة FD , SELECT معتادان .

ويجب ذكر أن برنامج الكوبل يمكن أن يحتوى على أى عدد من عبارات SORT ( أو MERGE ) ، وذلك بافتراض أنه لايوجد تداخل بين العبارات فقط .

بعد حصر العمل الشامل لعبارة SORT .. فإننا نوجه بقية هذا القسم إلى إلقاء نظرة فاحصة على أجزائها المنفصلة وصيغها المختلفة .

## جزء المفتاح تصاعدى أو تنازلى

الغرض من جزء المفتاح تصاعدى أو تنازلى ASCENDING / DESCENDING KEY ، هو تعريف مواقع المفاتيح داخل السجلات المراد ترتيبها . وبالنسبة للسجلات متغيرة الطول .. فمن المطلوب أن كل حقل مفتاح يتواجد فى نفس الموقع تماما داخل كل السجلات .

**مثال ١٢ - ٦ :**

اعتبر الملف التالى :

```
SD  PARTS-SORT-FILE
    RECORD CONTAINS 12 TO 22 CHARACTERS.
    .
01  MADE-PART-RECORD.
    05  MADE-PART-ID              PIC X(6).
    05  QUANTITY-ON-HAND          PIC S9(5)  COMP-3.
    05  MADE-WAREHOUSE-CODE       PIC X(3).
01  BOUGHT-PART-RECORD.
    05  BOUGHT-PART-ID            PIC X(6).
    05  BOUGHT-QUANTITY-ON-HAND   PIC S9(5)  COMP-3.
    05  BOUGHT-WAREHOUSE-CODE     PIC X(3).
    05  SUPPLIER-ID               PIC X(7).
    05  SUPPLIER-PRICE            PIC S9(3)V99  COMP-3.
```

طول السجل هو 12 بالنسبة إلى MADE - PART - RECORD ، و 22 بالنسبة الى BOUGHT - PART - RECORD، ، ولكنه – طبقا لما هو مطلوب – يحتل PART - ID نفس الموضع فى كل من السجلين (البايت من ١ – ٦ ) ، وكذلك هو الحال بالنسبة للحقل WAERHOUSE - CODE ؛ حيث يحتل الموضع من البايت ١٠ – ١٢ ، وعلى هذا .. يمكن أن تكتب عبارة SORT لهذا الملف على النحو التالى:

```
SORT PARTS-SORT-FILE
    ASCENDING KEY MADE-PART-ID
    DESCENDING KEY MADE-WAREHOUSE-CODE
    INPUT PROCEDURE IS SCREEN-PARTS
    OUTPUT PROCEDURE IS PRINT-REPORT
```

كما يمكن أن تأخذ الشكل التالى أيضا:

```
SORT PARTS-SORT-FILE
    ASCENDING KEY BOUGHT-PART-ID
    DESCENDING KEY BOUGHT-WAREHOUSE-CODE
    INPUT PROCEDURE IS SCREEN-PARTS
    OUTPUT PROCEDURE IS PRINT-REPORT
```

## جزء تسلسل التتابع

يتماثل جزء تسلسل التتابع collating sequence مع جزء PROGRAM COLLATING SEQUENCE، الذى سبق شرحه فى القسم الثالث من الفصل الرابع، إلا أن تسلسل التتابع المحدد لعبارة SORT ، لا يحتاج أن يكون هو نفسه مثل تسلسل التتابع المحدد لمقارنة عناصر حرفية عددية فى جزء الإجراءات. ويجب أن يعرف alphabet-name (شكل ١٢ - ١) فى مقطع الأسماء الخاصة (القسم الرابع الفصل الرابع). عادة يحذف تسلسل التتابع من عبارة SORT، مع اعتبار تسلسل التتابع التقليدى هو المستخدم.

## إجراء المدخلات والإزاحة، ...

## إجراء المخرجات والإعادة ...

عند استخدام إجراء المدخلات INPUT PROCEDURE .. يجب أن يحتوى على عبارة إزاحة RELEASE واحدة على الأقل .

<u>RELEASE</u> record-name [<u>FROM</u> data-name]

عبارة الإزاحة تشبه عبارة الكتابة WRITE ، ويمكن التفكير فيها فى واقع الأمر بأنها كتابة WRITE سجل ملف ترتيب فى منفعة الترتيب ؛ حيث تجمع كل هذه السجلات ، لترتيبها عندما ينتهى جزء إجراء المدخلات INPUT PROCEDURE ، ويعمل جزء FROM مثل جزء FROM المستخدم مع عبارة WRITE تماماً .

**مثال ١٢ - ٧ :**

فى مثال (١٢ - ٦) .. يمكن أن تشمل شفرة SCREEN - PARTS ما يلى:

```
IF MADE-PART
    RELEASE MADE-PART-RECORD
        FROM INPUT-RECORD
ELSE
    MOVE INPUT-ID TO BOUGHT-PART-ID
    MOVE INPUT-QUANTITY TO BOUGHT-QUANTITY-ON-HAND
    MOVE INPUT-CODE TO BOUGHT-WAREHOUSE-CODE
    MOVE INPUT-SUPPLIER TO SUPPLIER-ID
    MOVE INPUT-PRICE TO SUPPLIER-PRICE
    RELEASE BOUGHT-PART-RECORD
```

يجب أن يحتوى كل إجراء مخرجات OUTPUT PROCEDURE على عبارة إعادة RETURN واحدة على الأقل، التى تطلب – من منفعة الترتيب – وضع السجل المنطقى التالى بعد الترتيب فى منطقة سجل ملف الترتيب :

```
RETURN sort-file-name RECORD [INTO data-name]
     AT END imperative-statement
```

تشبه عبارة RETURN عبارة READ ، وتعمل RETURN... INTO مثل عمل READE ... INTO . أكثر من هذا .. يعمل جزء AT END (أى بعد انتهاء ملف الترتيب) مثل جزء AT END فى عبارة READ تماما .

## رتب ... معطيا ... مستخدما

عندما لا تكون هناك حاجة إلى تشغيل للسجلات المنطقية ، يسبق أو يتبع الترتيب .. استخدم رتب ... معطيا ... مستخدما SORT... GIVING... USING .

إلا أن الكفاءة تزداد كثيرا إذا ما نفذت منفعة الترتيب كبرنامج مستقل stand- alone pragram (بدلا من استدعائها من الكوبل). فى عبارة SORT من الكوبل، يجب إغلاق ملفات GIVING,USING عندما تنفذ عبارة SORT.

## رتب .. إجراء مدخلات ... معطيا ...

عند استخدام إجراء المدخلات INPUT PROCEDURE ، يجب تشكيل جزء الإجراءات كما فى مثال ١٢ . ٨ .

مثال ١٢ـ ٨ :

```
PROCEDURE DIVISION.
    SORT...
        ON...
        INPUT PROCEDURE IS SCREEN-INPUT
        OUTPUT PROCEDURE IS PRINT-REPORT
    STOP RUN

SCREEN-INPUT SECTION.
.................................
PRINT-REPORT SECTION.
.................................
END-SORT-PROCEDURES SECTION.
A-PARAGRAPH.
.................................
B-PARAGRAPH.
.................................
```

لاحظ أن فعل SORT وضع فى الجزء المنفذ ، الذى يليه مباشرة STOP RUN . ويعطى استخدام STOP RUN أثناء تنفيذ INPUT / OUTPUT PROCEDURE نتائج غير متوقعة ، وينتهى البرنامج نهاية غير طبيعية فى كوبل IBM OS / VS

لاحظ كذلك أن END - SORT - PROCEDURES SECTION ، بدون عنوان، ينتمى B- PARAGRAPH, A- PAR-AGRAPH إلى PRINT - REPORT SECTION ؛ فإذا كان البرنامج ينفذ مثلاً SCREEN - INPUT ، أو PRINT - REPORT أياً من المقطعين الخارجيين (كما هو مسموح به فى كوبل IBM OS/VS )، وعلى ذلك فالاختيار الأفضل للعنوان يمكن أن يكون : PERFORMED - PARAGRAPHS SECTION ، أو ما يشبهه. انظر المسألة ٣٢ من هذا الفصل للمزيد عن استخدام PERFORM فى INPUT / OUTPUT PROCEDURE .

ويمكن أن يكون أحد التطبيقات التقليدية لعبارة رتب ... إجراء مدخلات... معطيا SORT... INPUT PROCEDURE... GIVING فى برنامج تنقيح.

**مثال ١٢ - ٩ :**

نقح برنامج التنقيح الموجود فى المسألة ٢٢ – الفصل الحادى عشر ، والمسألة ٣٠ – الفصل الثامن بإدخال عبارة SORT .. تستخدم INPUT PROCEDURE فى غربلة السجلات غير الصحيحة ، وتستخدم جزء GIVING فى وضع العمليات الجارية الصحيحه والمرتبة فى ملف قرص.

انظر شكل (١٢ – ٢) مع تركيز الانتباه إلى هيكل جزء الإجراءات. جزء الإجراءات الأصلى (والمعطى جزء فقط منه فى شكل (١١ – ٥) أصبح إجراء مدخلات INPUT PROCEDURE لعبارة SORT . عبارة WRITE الأصلية (غير موجودة فى شكل ١١ – ٥ ، لكن انظر السطر ٢٠٩ فى شكل ٨ – ١٤، والتى حولت العمليات الجارية الصحيحة إلى القرص ) استبدلت بعبارة RELEASE والتى تعطى العمليات الجارية الصحيحة لمنفعة الترتيب ، والتى تنتج VALID - TIME - FILE كاستجابة لجزء GIVING . لاحظ أن ملف GIVING يجب أن يغلق ، عندما تنفذ SORT، حيث تفتح منفعة الترتيب OPEN ، وتغلق CLOSE ملفات GIVING, USING .

```
       INPUT-OUTPUT SECTION.
       FILE-CONTROL.

           SELECT TIME-CARD-FILE
               ASSIGN TO TIMECARD
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
           .
      *
      *    --------------------------------------------------
      *
      *    TO DO A COBOL SORT, ONE MUST DEFINE A SELECT STATEMENT
      *    FOR THE "SORT FILE", WHICH IS NOT ACTUALLY A FILE LIKE
      *    OTHER DATA FILES IN THE PROGRAM, BUT RATHER REPRESENTS
      *    THE AREA IN WHICH THE SORT/MERGE UTILITY WILL REARRANGE
      *    THE RECORDS TO BE SORTED.
      *
      *    --------------------------------------------------
      *
```

شكل (١٢ - ٢)

```
           SELECT SORTING-TIME-CARD-FILE
               ASSIGN TO SORTWK
           .
           SELECT VALID-TIME-FILE
               ASSIGN TO DISKTIME
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
           .
           SELECT ERROR-LISTING-FILE
               ASSIGN TO ERRORLOG
               ORGANIZATION IS SEQUENTIAL
               ACCESS IS SEQUENTIAL
           .

       DATA DIVISION.

       FILE SECTION.

       FD  TIME-CARD-FILE
           RECORD CONTAINS 80 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .

       01  TIME-CARD-RECORD.
           05  TIME-CARD-ID               PIC X(5).
           05  TIME-CARD-HOURS            PIC 9(2)V9.
           05  TIME-CARD-X-HOURS          REDEFINES TIME-CARD-HOURS
                                          PIC X(3).
           05  TIME-CARD-CLOSING-DATE.
               10  TIME-CARD-CLOSING-MO                   PIC 99.
               10  TIME-CARD-CLOSING-DAY                  PIC 99.
               10  TIME-CARD-CLOSING-YR                   PIC 99.
           05  TIME-CARD-DEPARTMENT       PIC X(4).
           05  FILLER                     PIC X(62).

      *    -------------------------------------------------
      *
      *    THE "SORT FILE" MUST BE DEFINED WITH AN "SD" RATHER
      *    THAN AN "FD".  THE "SD" DEFINES THE RECORD LAYOUT FOR
      *    THE RECORDS TO BE SORTED.  IT REPRESENTS THE MEMORY
      *    AREA FROM WHICH THE SORT/MERGE UTILITY PICKS UP ANY
      *    RECORDS TO BE SORTED, AND TO WHICH THE UTILITY RETURNS
      *    RECORDS (ONE AT A TIME) AFTER THEY HAVE BEEN REARRANGED.
      *
      *    -------------------------------------------------
      *
       SD  SORTING-TIME-CARD-FILE
           RECORD CONTAINS 23 CHARACTERS
           .

       01  VALID-TIME-RECORD.
           05  VALID-TIME-ID              PIC X(5).
           05  VALID-TIME-HOURS           PIC S9(2)V9    COMP-3.
           05  VALID-TIME-CLOSING-DATE    PIC X(6).
           05  VALID-TIME-DEPARTMENT      PIC X(4).
           05  VALID-TIME-EDIT-DATE       PIC X(6).

       FD  VALID-TIME-FILE
           BLOCK CONTAINS 0 RECORDS
           RECORD CONTAINS 23 CHARACTERS
           LABEL RECORDS ARE STANDARD
```

شكل ( ١٢ - ٢ ) تكملة

```


 01  SORTED-VALID-RECORD          PIC X(23).


 FD  ERROR-LISTING-FILE
     RECORD CONTAINS 132 CHARACTERS
     LABEL RECORDS ARE OMITTED
     .

 01  ERROR-REPORT-LINE            PIC X(132).

 WORKING-STORAGE SECTION.

 PROCEDURE DIVISION.

*    --------------------------------------------------
*
*    THE SORT VERB MUST NAME THE "SORT FILE" WHICH HAS BEEN
*    PREVIOUSLY DEFINED.  IT ALSO INDICATES THE KEY(S) ON
*    WHICH SORTING IS TO BE DONE (THE SORT KEY(S) MUST BE
*    PART OF THE SORT FILE RECORD DESCRIPTION), AND WHETHER
*    THE SORT IS TO BE ASCENDING/DESCENDING ON EACH FIELD.
*    IN THIS CASE THE ENTIRE EDIT PROGRAM IS MADE AN
*    INPUT PROCEDURE TO THE SORT.  THE SORTED RECORDS
*    WILL AUTOMATICALLY BE PLACED ON THE "GIVING" FILE.
*
*    --------------------------------------------------
*
     SORT SORTING-TIME-CARD-FILE
         ON ASCENDING KEY VALID-TIME-ID
         INPUT PROCEDURE IS 000-EDIT-TIME-CARDS
         GIVING VALID-TIME-FILE
     STOP RUN


 000-EDIT-TIME-CARDS SECTION.

     PERFORM 100-INITIALIZE
     PERFORM 200-PRODUCE-EDIT-LISTING
         UNTIL NO-MORE-RECORDS
     PERFORM 300-TERMINATE
     .

 PERFORMED-PARAGRAPHS SECTION.

 100-INITIALIZE.

     OPEN    INPUT   TIME-CARD-FILE
             OUTPUT  ERROR-LISTING-FILE
     MOVE "NO" TO                 WS-END-TIME-FILE-SW
     MOVE ZERO TO                 WS-PAGE-NUMBER
                                  WS-NUMBER-EMPLOYEES
                                  WS-TOTAL-HOURS
     MOVE WS-PAGE-SIZE TO         WS-LINES-ON-PAGE
     ACCEPT WS-SYSTEM-DATE FROM DATE
     MOVE SYSTEM-YY TO            WS-HEADING-YY
     MOVE SYSTEM-MM TO            WS-HEADING-MM
     MOVE SYSTEM-DD TO            WS-HEADING-DD
```

شكل ( ١٢ - ٢ ) تكملة

```
       200-PRODUCE-EDIT-LISTING.

           PERFORM 250-GET-TIME-CARD
           IF RECORD-AVAILABLE
               PERFORM 240-VALIDATE-FIELDS
               IF VALID-RECORD
                   ADD 1                TO WS-NUMBER-EMPLOYEES
                   ADD TIME-CARD-HOURS TO WS-TOTAL-HOURS
                   PERFORM 210-RELEASE-TO-SORT
               ELSE
                   PERFORM 220-PRODUCE-ERROR-LISTING


       210-RELEASE-TO-SORT.

      *    ---------------------------------------------------
      *
      *    RELEASE A RECORD TO THE SORT/MERGE UTILIITY FOR SORTING.
      *    THIS IS DONE BY MOVING THE RECORD INTO THE SORT FILE
      *    BUFFER AREA, THEN "RELEASING" THE RECORD.
      *
      *    ---------------------------------------------------

           MOVE TIME-CARD-ID            TO VALID-TIME-ID
           MOVE TIME-CARD-HOURS         TO VALID-TIME-HOURS
           MOVE TIME-CARD-CLOSING-DATE  TO VALID-TIME-CLOSING-DATE
           MOVE TIME-CARD-DEPARTMENT    TO VALID-TIME-DEPARTMENT
           MOVE WS-SYSTEM-DATE          TO VALID-TIME-EDIT-DATE
           RELEASE VALID-TIME-RECORD


       220-PRODUCE-ERROR-LISTING.

           MOVE TIME-CARD-ID            TO WS-DETAIL-ID
           MOVE TIME-CARD-X-HOURS       TO WS-DETAIL-HOURS
           MOVE TIME-CARD-CLOSING-DATE  TO WS-DETAIL-DATE
           MOVE TIME-CARD-DEPARTMENT    TO WS-DETAIL-DEPARTMENT

           MOVE WS-DETAIL-LINE          TO WS-OUTPUT-AREA
           MOVE WS-SKIP-BEFORE-DETAIL   TO WS-NUMBER-TO-SKIP
           PERFORM 230-PRINT-OUTPUT-AREA

           MOVE WS-UNDER-LINE           TO WS-OUTPUT-AREA
           MOVE WS-SKIP-BEFORE-UNDER    TO WS-NUMBER-TO-SKIP
           PERFORM 230-PRINT-OUTPUT-AREA
           .

       230-PRINT-OUTPUT-AREA.

           IF WS-LINES-ON-PAGE + WS-NUMBER-TO-SKIP
               GREATER THAN WS-PAGE-SIZE
                   ADD 1 TO WS-PAGE-NUMBER
                   MOVE WS-PAGE-NUMBER TO WS-HEADING-PAGE
                   WRITE ERROR-REPORT-LINE
                       FROM WS-HEADING-LINE-1
                       AFTER ADVANCING PAGE
                   WRITE ERROR-REPORT-LINE
                       FROM WS-HEADING-LINE-2
```

شكل (١٢ - ٢) تكملة

```
                        AFTER ADVANCING WS-SKIP-BEFORE-HEADING LINES
                    MOVE WS-SKIP-BEFORE-HEADING TO WS-LINES-ON-PAGE
        .
        WRITE ERROR-REPORT-LINE
            FROM WS-OUTPUT-AREA
            AFTER ADVANCING WS-NUMBER-TO-SKIP LINES
        ADD WS-NUMBER-TO-SKIP TO WS-LINES-ON-PAGE
        .

    240-VALIDATE-FIELDS.

        MOVE "NO"    TO WS-ERROR-DETECTED-SW
        MOVE SPACES TO WS-UNDER-LINE
        IF TIME-CARD-ID NOT NUMERIC
            MOVE ALL "-" TO WS-UNDER-ID
            MOVE "YES"   TO WS-ERROR-DETECTED-SW
        .
        IF TIME-CARD-HOURS NOT NUMERIC
            MOVE ALL "-" TO WS-UNDER-HOURS
            MOVE "YES"   TO WS-ERROR-DETECTED-SW
        .
        IF TIME-CARD-CLOSING-DATE NOT NUMERIC
            MOVE ALL "-" TO WS-UNDER-DATE
            MOVE "YES"   TO WS-ERROR-DETECTED-SW
        ELSE
            PERFORM 260-VALIDATE-DATE
        .
        IF TIME-CARD-DEPARTMENT NOT NUMERIC
            MOVE ALL "-" TO WS-UNDER-DEPARTMENT
            MOVE "YES"   TO WS-ERROR-DETECTED-SW


    250-GET-TIME-CARD.

        READ TIME-CARD-FILE
            AT END
                MOVE "YES" TO WS-END-TIME-FILE-SW


    260-VALIDATE-DATE.

        IF TIME-CARD-CLOSING-YR GREATER THAN SYSTEM-YY
            MOVE ALL "-" TO WS-UNDER-YR
            MOVE "YES" TO WS-ERROR-DETECTED-SW
        .
        IF TIME-CARD-CLOSING-MO LESS THAN 1 OR GREATER THAN 12
            MOVE ALL "-" TO WS-UNDER-MO
            MOVE "YES" TO WS-ERROR-DETECTED-SW
        ELSE IF TIME-CARD-CLOSING-DAY LESS THAN 1 OR
            GREATER THAN DAYS-PER-MONTH (TIME-CARD-CLOSING-MO)
                MOVE ALL "-" TO WS-UNDER-DAY
                MOVE "YES" TO WS-ERROR-DETECTED-SW


    300-TERMINATE.

        MOVE WS-NUMBER-EMPLOYEES     TO WS-FOOTING-COUNT
        MOVE WS-TOTAL-HOURS          TO WS-FOOTING-TOTAL
        MOVE WS-FOOTING-LINE         TO WS-OUTPUT-AREA
        MOVE WS-SKIP-BEFORE-FOOTING  TO WS-NUMBER-TO-SKIP
        MOVE WS-FOOTING-LINE         TO WS-OUTPUT-AREA
        PERFORM 230-PRINT-OUTPUT-AREA

        CLOSE    TIME-CARD-FILE
                 ERROR-LISTING-FILE
```

شكل ( ١٢ - ٢ ) تكملة

## رتب ... مستخدما ... إجراء مخرجات ...

يمكن استخدام رتب ... مستخدما ... إجراء مخرجات SORT... USING... OUTPUT PROCEDURE فى الترتيب، وإجراء تشغيل لمحتويات الملف بعد الترتيب، عندما لا تكون هناك حاجة إلى غربلة ، أو تشغيل سابق للسجلات المنطقية (مثل طباعة العناوين البريدية ) .

**مثال ١٢ – ١٠ :**

المسأله ٨ – ٣٢ الخاصة ببرنامج طباعة شيكات لنظام الرواتب ، والتى سبقت معالجة برنامج التنقيح الخاص بها فى المسألة ١١ – ٢٢ ، ومثال (١٢ – ٩) ، عدلها لطباعة الشيكات ، مرتبة ترتيبا أبجديا وطبقا لاسم العامل (بدلاً من ترتيبها طبقا لرقم العامل) .. ترتب السجلات فى PAYROLL - DISK - FILE .

انظر شكل (١٢ – ٣) ؛ حيث تشير سطور التعليقات من ١٥٠ إلى ١٦٠ إلى البرنامج الأصلى، الموجود فى خريطة الهيكل فى شكل (٨ – ١٧) ، لكنه غير معروض فى المسألة ٨ – ٣٢ .

```
00022           SELECT PAYROLL-DISK-FILE
00023               ASSIGN TO PAYDISK
00024               ORGANIZATION IS SEQUENTIAL
00025               ACCESS IS SEQUENTIAL
00026           .
00027           SELECT SORT-CHECK-FILE
00028               ASSIGN TO SORTUTIL
00029           .
00030           SELECT PAYCHECK-FILE
00031               ASSIGN TO CHECKS
00032               ORGANIZATION IS SEQUENTIAL
00033               ACCESS IS SEQUENTIAL
00034
00035
00036       DATA DIVISION.
00037
00038       FILE SECTION.
00039
00040       FD  PAYROLL-DISK-FILE
00041           BLOCK CONTAINS 0 RECORDS
00042           RECORD CONTAINS 80 CHARACTERS
00043           LABEL RECORDS ARE STANDARD
00044           .
00045
00046      *    -----------------------------------------------------
00047      *    NOTE THAT PAYROLL-CHECK-RECORD NEED NOT BE FULLY
00048      *    DESCRIBED WITH SORT...USING...
00049      *    -----------------------------------------------------
00050
00051       01  PAYROLL-CHECK-RECORD          PIC X(80).
00052
00053       SD  SORT-CHECK-FILE
00054           RECORD CONTAINS 80 CHARACTERS
00055           .
00056
00057       01  SORT-CHECK-RECORD.
00058           05  PAYROLL-ID                PIC X(5).
00059           05  PAYROLL-HOURS             PIC S9(2)V9.
00060           05  PAYROLL-DATE              PIC X(6).
```

شكل ( ١٢ – ٣ )

```
           05  PAYROLL-DEPARTMENT       PIC X(4).
           05  PAYROLL-RATE             PIC S9(3)V99.
           05  PAYROLL-NAME.
               10  PAYROLL-INITIALS     PIC X(4).
               10  PAYROLL-LAST-NAME    PIC X(16).
           05  PAYROLL-GROSS            PIC S9(6)V99.
           05  PAYROLL-TAX              PIC S9(5)V99.
           05  PAYROLL-NET              PIC S9(6)V99.
           05  FILLER                   PIC X(14).

       FD  PAYCHECK-FILE
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED
           .

       01  PAYCHECK-LINE                PIC X(132).

       WORKING-STORAGE SECTION.
.
.
.
       PROCEDURE DIVISION.

      *    ------------------------------------------------------
      *    THE SORT STATEMENT IS PLACED IN THE HIGHEST LEVEL
      *    MODULE.  THE REST OF THE ORIGINAL PROGRAM
      *    BECOMES THE OUTPUT PROCEDURE, WITH ONLY THE FOLLOWING
      *    CHANGES NECESSARY:  1) PAYROLL-DISK-FILE MUST NOT BE
      *    OPENED SINCE IT IS THE USING FILE, AND 2) "READ
      *    PAYROLL-DISK-FILE" BECOMES "RETURN SORT-CHECK-FILE"
      *    ------------------------------------------------------

           SORT SORT-CHECK-FILE
               ON ASCENDING KEY PAYROLL-LAST-NAME
               USING PAYROLL-DISK-FILE
               OUTPUT PROCEDURE IS PRINT-STUBS-AND-CHECKS
           STOP RUN
           .

       PRINT-STUBS-AND-CHECKS SECTION.

           PERFORM 100-INITIALIZE
           PERFORM 200-PRODUCE-PAYCHECKS
               UNTIL NO-MORE-RECORDS
           PERFORM 300-TERMINATE
           .
      *---------------------------------
       PERFORMED-PARAGRAPHS SECTION.
      *---------------------------------
       100-INITIALIZE.

           OPEN    OUTPUT  PAYCHECK-FILE
           MOVE "NO" TO                  WS-END-DISK-FILE-SW
           MOVE ZERO TO                  WS-NUMBER-EMPLOYEES
                                         WS-TOTAL-HOURS
           ACCEPT WS-TODAYS-DATE FROM DATE
           MOVE SYSTEM-YY                TO WS-YY
           MOVE SYSTEM-MM                TO WS-MM
           MOVE SYSTEM-DD                TO WS-DD
           ACCEPT WS-CHECK-NUMBER
               FROM STARTING-CHECK-NUMBER-DEVICE
```

شكل ( ١٢ - ٣ ) تكملة

```
           PERFORM 110-PRODUCE-TEMPLATE
           PERFORM 210-GET-NEXT-EMPLOYEE
           .

       110-PRODUCE-TEMPLATE.

           MOVE ALL "*" TO WS-STUB-ID
                           WS-STUB-NAME
                           WS-STUB-DATE
                           WS-CHECK-NAME
           PERFORM 230-PRINT-CHECK
               WS-NUMBER-OF-TEMPLATES TIMES
           .

       200-PRODUCE-PAYCHECKS.

           PERFORM 220-FORMAT-CHECKS
           PERFORM 230-PRINT-CHECK
           ADD PAYROLL-HOURS              TO WS-TOTAL-HOURS
           ADD 1                          TO WS-NUMBER-EMPLOYEES
                                             WS-CHECK-NUMBER
           PERFORM 210-GET-NEXT-EMPLOYEE
           .

       210-GET-NEXT-EMPLOYEE.

           RETURN SORT-CHECK-FILE
               AT END
                   MOVE "YES" TO WS-END-DISK-FILE-SW
           .

       220-FORMAT-CHECKS.

           MOVE PAYROLL-ID                TO WS-STUB-ID
           MOVE PAYROLL-NAME              TO WS-STUB-NAME
                                             WS-CHECK-NAME
           MOVE PAYROLL-DEPARTMENT        TO WS-STUB-DEPARTMENT
           MOVE PAYROLL-DATE              TO WS-STUB-DATE
           MOVE PAYROLL-HOURS             TO WS-STUB-HOURS
           MOVE PAYROLL-RATE              TO WS-STUB-RATE
           MOVE PAYROLL-GROSS             TO WS-STUB-GROSS
           MOVE PAYROLL-TAX               TO WS-STUB-TAX
           MOVE PAYROLL-NET               TO WS-STUB-NET
                                             WS-CHECK-AMOUNT


       230-PRINT-CHECK.

           WRITE PAYCHECK-LINE
               FROM WS-STUB-LINE-1
               AFTER ADVANCING WS-SKIP-BEFORE-STUB-1 LINES
           WRITE PAYCHECK-LINE
               FROM WS-STUB-LINE-2
               AFTER ADVANCING WS-SKIP-BEFORE-STUB-2 LINES
           WRITE PAYCHECK-LINE
               FROM WS-CHECK-LINE-1
               AFTER ADVANCING WS-SKIP-BEFORE-CHECK-1 LINES
           WRITE PAYCHECK-LINE
               FROM WS-CHECK-LINE-2
               AFTER ADVANCING WS-SKIP-BEFORE-CHECK-2 LINES


       300-TERMINATE.

           CLOSE   PAYCHECK-FILE
```

شكل ( ١٢ - ٣ ) تكملة

## رتب ... إجراء مدخلات ... إجراء مخرجات

**مثال ١٢ - ١١ :**

نوضح فى شكل (١٢ - ٤) كلاً من إجراء المدخلات وإجراء المخرجات مع برنامج إنتاج ملف مفهرس indexed file ، وهو ملف شائع الاستخدام جدا فى التطبيقات التى تحتاج إلى تشغيل ملفات عشوائيا. الطريقة الأكثر كفاءة لإنتاج ملفات مفهرسة هى بنائها تتابعيا، ووضع السجلات فى الملف؛ مرتبة طبقا لاحد المفاتيح. ويستخدم إجراء المدخلات فى اكتشاف سجلات العمليات الجارية غير الصحيحة ، ومنعها من الترتيب، ثم ينتج إجراء المخرجات بعد ذلك الملف المفهرس فعليا. لاحظ أن إجراء المدخلات وإجراء المخرجات يشبهان برنامجين منفصلين؛ كل منها يفتح ملفات ويضع قيماً ابتدائية لمفاتيح، ثم ينفذ مقطع تشغيل رئيسى ، حتى تنتهى البيانات، فيغلق الملفات وينهى العمل. ليس هذا مدهشا؛ لأن الجزءين ينفذان منفصلان عن بعضهما البعض تحت تحكم فعل SORT .

```
INPUT-OUTPUT SECTION.

FILE-CONTROL.
    .
    SELECT CREATION-INFO-FILE
        ASSIGN TO CREATE
        ORGANIZATION IS SEQUENTIAL
        ACCESS IS SEQUENTIAL
    .
    SELECT SORTED-TRANS-FILE
        ASSIGN TO SORTWK
    .
    SELECT PRICING-MASTER-FILE
        ASSIGN TO PRICES
        ORGANIZATION IS INDEXED
        ACCESS IS SEQUENTIAL
        RECORD KEY IS PRICING-ITEM-ID
        FILE STATUS IS MASTER-STATUS-CODE
    .
    SELECT ERROR-LOG
        ASSIGN TO ERRORS
        ORGANIZATION IS SEQUENTIAL
        ACCESS IS SEQUENTIAL
    .

DATA DIVISION.

FILE SECTION.

FD  CREATION-INFO-FILE
    RECORD CONTAINS 80 CHARACTERS
    LABEL RECORDS ARE OMITTED
    .
01  CREATION-RECORD.
    05  CREATION-ITEM-ID          PIC X(5).
    05  CREATION-ITEM-DESCRIPTION
                                  PIC X(25).
    05  CREATION-PRICE            PIC 9(5)V99.
    05  CREATION-PRICE-X          REDEFINES CREATION-PRICE
                                  PIC X(7).
    05  CREATION-QUANTITY         PIC 9(5).
    05  CREATION-QUANTITY-X       REDEFINES CREATION-QUANTITY
```

شكل (١٢ - ٤)

```
                                                PIC X(5).
             05  FILLER                         PIC X(38).

         SD  SORTED-TRANS-FILE
             RECORD CONTAINS 80 CHARACTERS
             .
         01  TRANS-RECORD.
             05  TRANSACT-ITEM-ID               PIC X(5).
             05  TRANSACT-ITEM-DESCRIPTION
                                                PIC X(25).
             05  TRANSACT-PRICE                 PIC S9(5)V99.
             05  TRANSACT-PRICE-X               REDEFINES TRANSACT-PRICE
                                                PIC X(7).
             05  TRANSACT-QUANTITY              PIC S9(5).
             05  TRANSACT-QUANTITY-X            REDEFINES TRANSACT-QUANTITY
                                                PIC X(5).
             05  FILLER                         PIC X(38).

         FD  PRICING-MASTER-FILE
             RECORD CONTAINS 37 CHARACTERS
             LABEL RECORDS ARE STANDARD
             .
         01  PRICING-RECORD.
             05  PRICING-ITEM-ID                PIC X(5).
           05  PRICING-ITEM-DESCRIPTION
                                              PIC X(25).
           05  PRICING-QUANTITY-ONHAND        PIC S9(5)       COMP-3.
           05  PRICING-PRICE                  PIC S9(5)V99    COMP-3.

       FD  ERROR-LOG
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED
           LINAGE IS 60 LINES
               WITH FOOTING AT 59
               LINES AT TOP 3
               LINES AT BOTTOM 3


       01  ERROR-LINE                         PIC X(132).

       WORKING-STORAGE SECTION.

       01  PROGRAM-SWITCHES.
           05  WS-END-CREATION-SW             PIC X(3).
               88  NO-MORE-RECORDS            VALUE "YES".
               88  MORE-RECORDS               VALUE "NO".
           05  WS-VALID-TRANS-SW              PIC X(3).
               88  VALID-TRANS                VALUE "YES".
               88  INVALID-TRANS              VALUE "NO".
           05  MASTER-STATUS-CODE             PIC XX.

       01  PROGRAM-COUNTERS.
           05  WS-PAGE-NUMBER                 PIC S9(3)       COMP-3.

       01  SYSTEM-DATE-AREA.
           05  SYSTEM-YY                      PIC 99.
           05  SYSTEM-MM                      PIC 99.
           05  SYSTEM-DD                      PIC 99.
```

شكل ( ١٢ - ٤ ) تكملة

```
       01  PROGRAM-CONSTANTS.
           05  WS-MESSAGES.
               10  WS-INVALID-FIELDS-MSG           PIC X(30)
                                       VALUE "INVALID FIELDS".
               10  WS-FILE-FULL-MSG                PIC X(30)
                                       VALUE "FILE FULL--NO ADD".
               10  WS-DUPLICATE-MESSAGE            PIC X(30)
                                       VALUE "DUPLICATE KEY--NO ADD".


       01  WS-HEADING-LINE-1.
           05  FILLER                  PIC X(3)        VALUE SPACES.
           05  FILLER                  PIC X(32)
                             VALUE "PRICING FILE CREATION ERROR LOG".
           05  WS-MM                   PIC Z9.
           05  FILLER                  PIC X           VALUE "/".
           05  WS-DD                   PIC 99.
           05  FILLER                  PIC X           VALUE "/".
           05  WS-YY                   PIC 99.
           05  FILLER                  PIC X(2)        VALUE SPACES.
           05  FILLER                  PIC X(5)        VALUE "PAGE ".
           05  WS-HEADING-PAGE         PIC ZZ9.
           05  FILLER                  PIC X(79)       VALUE SPACES.

       01  WS-HEADING-LINE-2.
           05  FILLER                  PIC X(10)       VALUE "ITEM-ID".
           05  FILLER                  PIC X(27)
                                       VALUE "DESCRIPTION".
           05  FILLER                  PIC X(9)        VALUE "PRICE".
           05  FILLER                  PIC X(86)       VALUE "QUANTITY".

        01  WS-DETAIL-LINE.
            05  WS-DETAIL-ITEM-ID       PIC X(5).
            05  FILLER                  PIC X(5)        VALUE SPACES.
            05  WS-DETAIL-DESCRIPTION   PIC X(25).
            05  FILLER                  PIC X(2)        VALUE SPACES.
            05  WS-DETAIL-PRICE         PIC X(7).
            05  FILLER                  PIC X(2)        VALUE SPACES.
            05  WS-DETAIL-QUANTITY      PIC X(5).
            05  FILLER                  PIC X(2)        VALUE SPACES.
            05  WS-DETAIL-MESSAGE       PIC X(79).

        PROCEDURE DIVISION.

            SORT SORTED-TRANS-FILE
                ON ASCENDING KEY TRANSACT-ITEM-ID
                INPUT PROCEDURE IS 000-SCREEN-TRANSACTIONS
                OUTPUT PROCEDURE IS 001-CREATE-PRICING-MASTER
            STOP RUN


        000-SCREEN-TRANSACTIONS SECTION.

            OPEN OUTPUT ERROR-LOG
            OPEN INPUT  CREATION-INFO-FILE
            MOVE "NO" TO WS-END-CREATION-SW
```

شكل ( ١٢ - ٤ ) تكملة

```
          ACCEPT SYSTEM-DATE-AREA  FROM DATE
          MOVE SYSTEM-MM   TO WS-MM
          MOVE SYSTEM-DD   TO WS-DD
          MOVE SYSTEM-YY   TO WS-YY

          MOVE ZERO        TO WS-PAGE-NUMBER

          PERFORM 170-PRINT-HEADINGS
          PERFQRM 180-GET-NEXT-TRANS

          PERFORM 100-GENERATE-PRICING-RECORD
              UNTIL NO-MORE-RECORDS

          CLOSE   CREATION-INFO-FILE


      001-CREATE-PRICING-MASTER SECTION.

          OPEN OUTPUT PRICING-MASTER-FILE
          MOVE "NO" TO WS-END-CREATION-SW
          PERFORM 140-RETURN-NEXT-TRANS
          PERFORM 120-PRODUCE-MASTER-REC
              UNTIL NO-MORE-RECORDS
          CLOSE   PRICING-MASTER-FILE
                  ERROR-LOG


      PERFORMED-PARAGRAPHS SECTION.

      100-GENERATE-PRICING-RECORD.

          PERFORM 110-VALIDATE-TRANS
          IF VALID-TRANS
              PERFORM 125-RELEASE-TO-SORT
          ELSE
              MOVE WS-INVALID-FIELDS-MSG  TO WS-DETAIL-MESSAGE
              PERFORM 130-PRODUCE-ERROR-LOG
          .
          PERFORM 180-GET-NEXT-TRANS
          .

      110-VALIDATE-TRANS.

          IF      CREATION-PRICE NOT NUMERIC
              OR CREATION-QUANTITY NOT NUMERIC
                  MOVE "NO" TO WS-VALID-TRANS-SW
          ELSE
              MOVE "YES" TO WS-VALID-TRANS-SW
          .

      120-PRODUCE-MASTER-REC.

          MOVE TRANSACT-ITEM-ID    TO PRICING-ITEM-ID
          MOVE TRANSACT-ITEM-DESCRIPTION
              TO PRICING-ITEM-DESCRIPTION
          MOVE TRANSACT-PRICE      TO PRICING-PRICE
          MOVE TRANSACT-QUANTITY   TO PRICING-QUANTITY-ONHAND
          PERFORM 150-WRITE-MASTER-RECORD
          IF MASTER-STATUS-CODE NOT EQUAL "00"
```

شكل ( ١٢ - ٤ ) تكملة

```
               MOVE WS-DUPLICATE-MESSAGE    TO WS-DETAIL-MESSAGE
               PERFORM 130-PRODUCE-ERROR-LOG
           .
           PERFORM 140-RETURN-NEXT-TRANS
           .

       125-RELEASE-TO-SORT.

           RELEASE TRANS-RECORD
               FROM CREATION-RECORD
           .

       130-PRODUCE-ERROR-LOG.

           MOVE TRANSACT-ITEM-ID          TO WS-DETAIL-ITEM-ID
           MOVE TRANSACT-ITEM-DESCRIPTION
                                          TO WS-DETAIL-DESCRIPTION
           MOVE TRANSACT-PRICE-X          TO WS-DETAIL-PRICE
           MOVE TRANSACT-QUANTITY-X       TO WS-DETAIL-QUANTITY
           PERFORM 160-PRINT-DETAIL-LINE
           .

       140-RETURN-NEXT-TRANS.

           RETURN SORTED-TRANS-FILE
               AT END
                   MOVE "YES" TO WS-END-CREATION-SW
           .


       150-WRITE-MASTER-RECORD.

           WRITE PRICING-RECORD
           .

       160-PRINT-DETAIL-LINE.

           WRITE ERROR-LINE
               FROM WS-DETAIL-LINE
               AFTER ADVANCING 2 LINES
               AT END-OF-PAGE
                   PERFORM 170-PRINT-HEADINGS
           .

       170-PRINT-HEADINGS.

           ADD 1 TO WS-PAGE-NUMBER
           MOVE WS-PAGE-NUMBER            TO WS-HEADING-PAGE
           WRITE ERROR-LINE
               FROM WS-HEADING-LINE-1
               AFTER ADVANCING PAGE
           WRITE ERROR-LINE
               FROM WS-HEADING-LINE-2
               AFTER ADVANCING 2 LINES
           .

       180-GET-NEXT-TRANS.

           READ CREATION-INFO-FILE
               AT END
                   MOVE "YES" TO WS-END-CREATION-SW
```

شكل (١٢ - ٤) تكملة

## ترتيب مجموعة عناصر

يمكن أن تتحسن كفاءة الترتيب طبقا لعدة مفاتيح بتجميع حقول المفاتيح مع بعضها البعض، وإجراء الترتيب طبقا لمجموعة العناصر. ويمكن تطبيق هذه الطريقة عندما تكون المفاتيح المطلوب تجميعها:

(١) متجاورة داخل السجل.

(٢) حرفية عددية أو عددية بدون إشارة مع استخدام DISPLAY .

(٣) كلها تصاعدية أو كلها تنازلية.

(٤) مرتبة من الحقل الرئيسى الى الحقل الادنى ؛ حيث يكون الحقل الرئيسى أول عنصر فى المجموعة ، والحقل الأدنى أخر عنصر فى المجموعة.

مثال ١٢ - ١٢ :

```
SD  SORT-FILE . . .
01  SORT-RECORD.
    05  SORT-KEYS.
        10  REGION-ID          PIC X(4).
        10  DEPARTMENT-ID      PIC X(7).
        10  SALESPERSON-ID     PIC X(3).
    05  REGION-NAME            PIC X(10).
    05  DEPARTMENT-NAME        PIC X(20).
    05  SALESPERSON-NAME       PIC X(15).
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
    SORT SORT-FILE
        ON ASCENDING KEY SORT-KEYS
. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
```

## ١٢ - ٣ دمج الملفات بفعل الدمج

الدمج merging هو خلط ملفين مرتبين فعلا (أو أكثر) مع بعضهما فى ملف واحد ، ويمكن الكلام عن دمج n طريقة (n-way merge) للإشارة الى دمج n ملف فى ملف واحد . وفى كوبل IBM OS/VS .. يجب أن تكون $n \leq 8$ . وتعالج نفس منفعة النظام التى تجرى ترتيب الملفات ، فى العادة - دمج الملفات.

وحيث يشمل كل من الترتيب والدمج ترتيبا ordering للسجلات فى ملفات .. فإن مفهوم الحقول الرئيسية ، أو حقول المفاتيح والتتابع التصاعدى أو التنازلى، والمفاتيح : الرئيسى والمتوسط والأدنى هى نفسها فى كلتا الحالتين.

يشبه تكوين عبارة الدمج MERGE شكل (١٢ - ٥) تكوين عبارة الترتيب . ولعمل أى منهما .. يجب أن يعرف المبرمج ملف ترتيب ، أو دمجاً خاصاً بالصورة المقيدة لعبارة SELECT فى جزء الأوساط ، وباستخدام SD فى جزء البيانات. وبعيدا عن شكل عبارات SORT,MERGE نفسها .. فإن متطلبات الكوبل لعمل الترتيب والدمج لا تتغير .

يجب أن تعرف عبارة الدمج MERGE حقول المفاتيح التى رتبت على أساسها الملفات ، المراد دمجها ، والتى على أساسها يرتب الملف المدمج. وتكتب المفاتيح فى تسلسل من المفتاح الرئيسى (أولا) الى المفتاح الأدنى (أخيرا)، كما هو الحال تماما فى عبارة SORT .

```
MERGE file-name-1
    ON {ASCENDING | DESCENDING} KEY data-name-1 [data-name-2] . . .
   [ON {ASCENDING | DESCENDING} KEY data-name-3 [data-name-4] . . .]
   [COLLATING SEQUENCE IS alphabet-name]
    USING file-name-2 file-name-3 [file-name-4] . . .
   {GIVING file-name-5
   {OUTPUT PROCEDURE IS section-name-1 [{THROUGH | THRU} section-name-2]}
```

شكل ( ١٢ - ٥ )

لاحظ أنه غير مسموح باستخدام INPUT PROCEDURE فى عبارة MERGE . يجب أن تغلق ملفات USING (المراد دمجها طبقا لمواصفات ASCENDING / DESCENDING KEY ) . عندما تنفذ عبارة MERGE ؛ حيث تفتحها عبارة MERGE تلقائيا . وعندما تنتهى عملية الدمج .. تغلق MERGE الملفات تلقائيا.

يحدد جزء GIVING أن السجلات المدمجة تكتب فى الملف المحدد مباشرة ، و يجب أن يكون ملف GIVING مغلقا عندما تنفذ MERGE ؛ حيث تفتح منفعة الترتيب أو الدمج الملفات تلقائيا، وتكتب السجلات المدمجة فى الملف، ثم تغلق الملف مرة أخرى.

يعمل إجراء المخرجات OUTPUT PROCEDURE مثل نظيره فى ملف عبارة SORT تماما.

| *File A* | *File B* | *File C* | *Merged File* |
|---|---|---|---|
| 20 | 10 | 5 | 5 |
| 40 | 30 | 25 | 10 |
| 60 | 50 | 35 | 20 |
| 80 | 70 | | 25 |
| | 90 | | 30 |
| | | | 35 |
| | | | 40 |
| | | | 50 |
| | | | 60 |
| | | | 70 |
| | | | 80 |
| | | | 90 |

شكل ( ١٢ - ٦ )

مثال ١٢ - ١٣ :

لدمج الملفات المبينة فى شكل ( ٢ - ٦ ) .. فإننا نكتب الشفرة التالية :

```
ENVIRONMENT DIVISION.
. . . . . . . . . . . . . . . . . . . . . . .
    SELECT FILE-A ASSIGN TO...
    SELECT FILE-B ASSIGN TO...
    SELECT FILE-C ASSIGN TO...
    SELECT MERGE-FILE
        ASSIGN TO SORTWK.
. . . . . . . . . . . . . . . . . . . . . . .

DATA DIVISION.

FILE SECTION.

FD  FILE-A...
01  RECORD-A...

FD  FILE-B...
01  RECORD-B...

FD  FILE-C
01  RECORD-C...

SD  MERGE-FILE
    RECORD CONTAINS...
01  MERGE-RECORD.
    05  MERGE-KEY-1...
    05  DATA-FIELD-1...
    05  MERGE-KEY-2...
    05  DATA-FIELD-2...
. . . . . . . . . . . . . . . . . . . . . . . . . . .
PROCEDURE DIVISION.

    MERGE MERGE-FILE
        ON ASCENDING KEY MERGE-KEY-1
        ON DESCENDING KEY MERGE-KEY-2
        USING FILE-A  FILE-B  FILE-C
        OUTPUT PROCEDURE IS PROCESS-MERGED-RECORDS
    STOP RUN

PROCESS-MERGED-RECORDS SECTION.
. . . . . . . . . . . . . . . . . . . . . . . . . . .
PERFORMED-PARAGRAPHS SECTION.
GET-MERGED-RECORD.
    RETURN MERGE-FILE
        AT END...
```

يظل هذا البرنامج صحيحا مع استبدال عبارة MERGE بعبارة SORT ، يمكن أن يكون هذا الاستبدال ضروريا إذا لم تكن الملفات المراد ترتيبها مرتبة فعلا ، وحيث إنها مرتبة .. فيصبح فعل MERGE أكثر كفاءة بالطبع عن فعل SORT .

لاحظ أنه إذا تحدد أكثر من ملف USING واحد فى عبارة SORT .. تخلط كل السجلات من كل الملفات وترتب. وعلى هذا.. يكون لدينا أسلوب لدمج ملفات غير مرتبة.

## اسئلة مراجعة

١٢ – ١ ميز بين ترتيب ملف وترتيب جدول.

١٢ – ٢ وضح المقصود : حقل مفتاح، ترتيب تصاعدى، ترتيب تنازلى ، مفتاح رئيسى، مفتاح متوسط، مفتاح أدنى.

١٢ – ٣ وضح العلاقة بين منفعة الترتيب ومنفعة الدمج للنظام من ناحية ، وعبارة SORT وعبارة MERGE من ناحية أخرى .

١٢ – ٤ ما المقصود بملف الترتيب ؟

١٢ – ٥ اذكر قيود عبارة SELECT وأجزاء FD عند تعريف ملف ترتيب ، أو ملف دمج ..

١٢ – ٦ وضح الغرض من عبارة RELEASE وعبارة RETURN .

١٢ – ٧ اذكر عدة أمثلة يمكن أن يكون فيها إجراء المدخلات مفيدا.

١٢ – ٨ اذكر عدة أمثلة يمكن أن يكون فيها إجراء المخرجات مفيدا.

١٢ – ٩ اذكر مثالاً يمكن أن يبنى إجراء المدخلات فيه السجلات التى يزيحها إلى منفعة الترتيب أو الدمج.

١٢ – ١٠ اذكرقيود مفاتيح الترتيب عندما ترتب سجلات متغيرة الطول.

١٢ – ١١ وضح الغرض من جزء تسلسل التتابع فى عبارات SORT,MERGE .

١٢ – ١٢ لماذا لا يشيع ترتيب SORT... GIVING فى برامج الكوبل ؟

١٢ – ١٣ ناقش هيكل جزء الإجراءات عند استخدام عبارة SORT ، أو عبارة MERGE .

١٢ – ١٤ ناقش سلسلة الأحداث التى تحدث عند تنفيذ عبارة SORT .

١٢ – ١٥ ماذا يحدث إذا نفذت العبارة STOP RUN داخل إجراء مدخلات أو داخل إجراء مخرجات ؟

١٢ – ١٦ ناقش العلاقة بين إجراء المدخلات وإجراء المخرجات تحت تحكم أمر SORT .

١٢ – ١٧ وضح كيف يحسن تجميع مفاتيح الترتيب من كفاءة الترتيب. ما القيود التى تقع على هذا الأسلوب ؟

١٢ – ١٨ ميز بين ترتيب الملف ودمج الملفات.

١٢ – ١٩ اذكر بعض الأمثلة للمواقف التى يمكن فيها استخدام دمج الملفات .

١٢ – ٢٠ ماذا يعنى دمج ثلاث طرق three - way merge ؟

١٢ – ٢١ لماذا يجب إغلاق ملفات GIVING, USING ، قبل تنفيذ SORT ، أو MERGE ؟

١٢ – ٢٢ كيف تبرمج عملية الدمج إذا لم تكن الملفات المطلوب دمجها مرتبة ؟

## مسائل محلولة

١٢ – ٢٣ (أ) عرف المفتاح الرئيسى ، والمتوسط ، والأدنى فى حقول السجل التالى.

(ب) أى المفاتيح يكون تصاعديا ، وأيها يكون تنازليا ؟

| *Name* | *Region* | *Number-of-Accounts* |
|---|---|---|
| PERELMAN | 10 | 5 |
| SNYDER | 10 | 5 |
| ABEL | 10 | 3 |
| CONVERSE | 10 | 3 |
| FOLKERS | 20 | 8 |
| GETZ | 20 | 8 |
| BAKER | 20 | 4 |

(أ) الحقل الرئيسى : region

الحقل المتوسط : number - of - accounts

الحقل الأدنى : name

(ب) name, Region فى ترتيب تصاعدي ، بينما number - of - accounts فى ترتيب تنازلى.

١٢ - ٢٤ بالإضافة إلى منفعة الترتيب والدمج.. ما برامج المنفعة الأخرى المتاحة تقليديا فى نظام الكمبيوتر ؟

برامج لنسخ الملفات، ولطباعة محتويات ملف، ولسرد كل الملفات على مجموعة أقراص معينة أو على شريط ؛ لمحاولة استعادة ملفات من مجموعة أقراص ، أو شريط أصابها التلف ... إلخ .

١٢ - ٢٥ ما الفرق بين تعريفات الكوبل لملف ترتيب وملف دمج ؟

لا يوجد فرق ؛ إذ يحتاج كل منهما إلى صيغة محددة لعبارة SELECT ، ويعرف كل منهما على مستوى SD .

لا توجد طريقة تذكر إذا كان ملف الترتيب مستخدما مع SORT ، أو مع MERGE ، وذلك باستثناء فحص العبارات الفعلية لجزء الإجراءات.

١٢ - ٢٦ ما الصيغة المناسبة لاسم ASSIGN TO لملف ترتيب أو دمج ؟

أى اسم صحيح ؛ حيث يعامل اسم ASSIGN TO كتعليق .

١٢ - ٢٧ اذكر ما إذا كان ما يلى يشمل : USING ، أو INPUT PROCEDURE ، أو GIVING ، أو OUTPUT PRO-CEDURE :

(أ) برنامج يطبع قائمة بعناوين بريدية لكل السجلات الموجودة فى شريط عناوين العملاء البريدية.

(ب) برنامج يحسب ويطبع متوسط نقاط درجات الطلبة الدارسين (دون الحصول على مؤهل جامعى) بأحد الجامعات.

(جـ) برنامج يعد نسخة احتياطية من ملف شيكات حسابات الدائنين ، يجب أن يرتب طبقا لرقم دفتر الأستاذ العام ؛ بدلا من ترتيبه طبقاً لرقم الشيك.

(د) برنامج تنقيح ينتج ملف قرص لعمليات جارية مرتبة ، وذلك بعد التأكد من صحتها.

(أ) USING مع OUTPUT PROCEDURE ( لتشكيل تقرير الطباعة) .

(ب) INPUT PROCEDURE ( لاختيار الطلبة غير الدراسين ؛ للحصول على مؤهل جامعى ) مع OUTPUT PROCE-DURE ( لحساب متوسط النقاط ) .

(جـ) USING مع GIVING ( فى الحياة الواقعية .. يمكن تشغيل منفعة الترتيب أو الدمج كبرنامج قائم بذاته ؛ بدلا من استدعائه من خلال برنامج الكوبل) .

(د) INPUT PROCEDURE (لغربلة العمليات الجارية غير الصحيحة ) مع GIVING .

١٢ – ٢٨ حدد الخطأ فيما يلى :

```
SD  ANY-SORT-FILE
    BLOCK CONTAINS 10 RECORDS
    RECORD CONTAINS 200 CHARACTERS
    LABEL RECORDS ARE STANDARD
```

يسمح بالأجزاء RECORD CONTAINS , DATA RECORDS ARE فقط فى تعريف ملف الترتيب أو الدمج (SD).
لا يعد الجزءان BLOCK CONTAINS, LABEL RECORDS صحيحين .

١٢ – ٢٩ اكتب برنامجا يرتب سجلات مبيعات ؛ طبقا لمتوسط المبيعات للعميل ، كما يحسب من إجراء المدخلات .

```
SELECT SALES-FILE ASSIGN TO...
SELECT SORT-FILE ASSIGN TO ANYTHING.
.....................................
FD  SALES-FILE...
01  SALES-RECORD.
    05  SALESPERSON-ID           PIC X(7).
    05  TOTAL-SALES              PIC S9(5)V99   COMP.
    05  NUMBER-CUSTOMERS         PIC S9(3)      COMP.
SD  SORT-FILE
    RECORD-CONTAINS...
01  SORT-RECORD.
    05  SORT-SALES-ID            PIC X(7).
    05  SORT-TOTAL-SALES         PIC S9(5)V99   COMP.
    05  SORT-NUMBER-CUSTOMERS    PIC S9(3)      COMP.
    05  SORT-AVERAGE-SALES       PIC S9(5)V99   COMP.
.............................
SORT SORT-FILE
   ON DESCENDING KEY SORT-AVERAGE-SALES
   INPUT PROCEDURE IS CALCULATE-AVERAGE
   OUTPUT PROCEDURE IS PRINT-REPORT
STOP RUN

CALCULATE-AVERAGE SECTION.
    OPEN  INPUT  SALES-FILE
    MOVE "NO" TO END-FILE-SW
    PERFORM GET-SALES-RECORD
    PERFORM RELEASE-A-RECORD
        UNTIL END-FILE-SW EQUAL "YES"
```

```
     CLOSE SALES-FILE
     .
 PRINT-REPORT SECTION.
 . . . . . . . . . . . . . . . . . . . . . . . . . . .
 PERFORMED-PARAGRAPHS SECTION.

 GET-SALES-RECORD.
     READ SALES-FILE
         AT END  MOVE "YES" TO END-FILE-SW
     .
 RELEASE-A-RECORD.
     MOVE SALESPERSON-ID       TO SORT-SALES-ID
     MOVE TOTAL-SALES          TO SORT-TOTAL-SALES
     MOVE NUMBER-CUSTOMERS  TO SORT-NUMBER-CUSTOMERS
     DIVIDE TOTAL-SALES BY NUMBER-CUSTOMERS
         GIVING SORT-AVERAGE-SALES
     RELEASE SORT-RECORD
     PERFORM GET-SALES-RECORD
     .
```

١٢ - ٣٠ يضع تسلسل التتابع المعتاد EBCDIC الأرقام من 0 إلى 9 بعد الحروف من A الى Z . قم بعمل ترتيب ؛ يضع الأرقام من 0 إلى 9 قبل الحروف من A إلى Z .

```
. . . . . . . . . . . . . . . . . . . . . . . . . .
SPECIAL-NAMES.
    CUSTOMER-SORT-ORDER IS "0" THRU "9"
                           "A" THRU "Z"
    . . . . . . . . . . . . . . . . . . . . . . .
    SORT SORT-FILE
        ON ASCENDING KEY WHATEVER
        COLLATING SEQUENCE IS CUSTOMER-SORT-ORDER
        INPUT PROCEDURE IS SCREEN-THEM
        OUTPUT PROCEDURE IS PROCESS-THEM
```

١٢ - ٣١ ناقش كيف يمكن ترتيب حقول تاريخ .

يمكن أن ترتب التواريخ المعرفة بالصورة (6) PIC X أو (6) PIC 9 بطريقة صحيحة إذا كانت الستة رموز فى حقل التاريخ مرتبة على النحو التالى yymmdd (yy هى السنة ، و mm الشهر ، و dd اليوم ) بدلا من الشكل المعتاد mmddyy . يحتفظ الشكل yymmdd بالترتيب المناسب لمفاتيح الترتيب من الرئيسى إلى الأدنى داخل حقل التاريخ. والطريقة الأخرى هى

استخدام التاريخ برقم اليوم فى السنة yyddd . ( القسم السادس - الفصل السادس ) . لا تنتج أى من هاتين الطريقتين نتائج صحيحة إذا تغير القرن.

١٢ - ٣٢ لا تسمح بعض مترجمات الكوبل بإجراء مدخلات ، أو إجراء مخرجات لتنفيذ مقاطع ، تقع خارج قسم المدخلات أو قسم المخرجات المناسب .

بافتراض استخدام مثل هذا المترجم .. أعد كتابة إجراء مدخلات المسألة ٢٩ .

```
CALCULATE-AVERAGE SECTION.
    OPEN INPUT SALES-FILE
    MOVE "NO" TO END-FILE-SW
    PERFORM GET-SALES-RECORD
    PERFORM RELEASE-A-RECORD
        UNTIL END-FILE-SW EQUAL "YES"
    CLOSE SALES-FILE
    GO TO CALCULATE-AVERAGE-EXIT
    .
GET-SALES-RECORD.
.................................
RELEASE-A-RECORD.
.................................
CALCULATE-AVERAGE-EXIT.
    EXIT.
PRINT-REPORT SECTION.
.................................
```

تعد عبارة GO TO عبارة واسعة الاستخدام فى كوبل غير المرتب ، وتكون هناك حاجة لهذه العبارة فى إعداد الشفرة السابقة لمنع GET - SALES - RECORD ، و RELEASE - A - RECORD الموجودين الآن فى SECTION من التنفيذ مره اخرى بعد إغلاق SALES - FILE . لا تفعل عبارة EXIT شيئاً ، ولكنها تسمح بأن يعمل CALCULATE - AVERAGE - EXIT كمقطع إنهاء القسم SECTION .

١٢ - ٣٣ أعد تعريف سجل الترتيب التالى لعمل ترتيب أكثر كفاءة .

```
SD  SORT-FILE...
01  SORT-REC.
    05  EMPLOYEE-ID          PIC X(4).
    05  NAME                 PIC X(20).
    05  DEPT                 PIC XX.
    05  NUMBER-DEPENDENTS    PIC S9        COMP-3.
    05  HOURLY-RATE          PIC S9(2)V99  COMP.
.........................
    SORT SORT-FILE
        ON DESCENDING KEY HOURLY-RATE
        ON ASCENDING KEY DEPT
        ON ASCENDING KEY EMPLOYEE-ID
```

```
SD   SORT-FILE...
01   SORT-REC.
     05  SORT-KEYS.
         10  DEPT                  PIC XX.
         10  EMPLOYEE-ID           PIC X(4).
     05  NAME                      PIC X(20).
     05  NUMBER-DEPENDENTS         PIC S9        COMP-3.
     05  HOURLY-RATE               PIC S9(2)V99  COMP.
...........................
     SORT SORT-FILE
         ON DESCENDING KEY HOURLY-RATE
         ON ASCENDING KEY SORT-KEYS
```

١٢ - ٣٤ كيف يدمج الملفان التاليان :

File A: 10, 5, 18, 17, 23 File B: 20, 30, 40, 50

حيث إن الملف A غير مرتب .. فيجب أن يدمج الملفان باستخدام عبارة SORT :

```
SORT SORT-FILE
   ON...
   USING  FILE-A  FILE-B
   OUTPUT PROCEDURE IS...
```

١٢ - ٣٥ حدد الخطأ فيما يلي :

```
PROCEDURE DIVISION.
    OPEN  INPUT  FILE-A
                 FILE-B
    MERGE MERGE-WORK-FILE
        ON ASCENDING KEY MERGE-KEY
        USING FILE-A
              FILE-B
        OUTPUT PROCEDURE IS PROCESS-RECORDS
```

يتطلب جزء USING أن يغلق ملف USING عند تنفيذ عبارة SORT ، أو عبارة MERGE (حيث تفتحه منفعة الترتيب أو الدمج تلقائيا ) .

# تمارين برمجة

١٢ – ٣٦ اكتب برنامج مطالبة لوكيل تجميع ديون:

• **المدخلات** : بطاقـة تحتوى على رقم الحساب ( الأعمدة١ – ٦ ) ، الاسم ( الأعمدة ١١ – ٣٠ ) ، المبلغ المدان به (الأعمدة ٣١ – ٣٧ ، مع تخصيص خانتين للكسر العشرى ) .

• **المخرجات** : تبدأ طباعة تقرير فى صفحة جديدة وله عنوان ، و عناوين أعمدة مناسبة ، مع طباعة رقم الحساب والا والمبلغ المدان به فى الصفحة . قم بعمل التخطيط الذى تراه مناسبا للتقرير ، و رتب الحسابات ترتيبا تنازليا ؛ طبقا للمب المدان به العميل. اطبع فى النهاية إجمالى عدد العملاء ، وإجمالى المبالغ المدانين بها .

١٢ – ٣٧ اكتب برنامجا لتساعد الجامعة فى تتبع الطبة الذين لم يدفعوا ثمن المذكرات .

• **المدخلات** : بطاقات بها رقم تعريف الطالب ( فى الصورة X(9) ) واسمه ( فى الصورة X(20) ) ، والمبلغ المدان به الطالب (فى صورة 9(2)V9(2) ) .

• **المخرجات** : طباعة تقرير له عنوان ، وعناوين أعمدة مناسبة . اطبع معلومات الطالب عبر الصفحة ، ويجب أن يسرد التقرير الطلبة ؛ بترتيب الطالب صاحب أكبر دين اولا ، و هكذا .. حتى يأتى الطالب صاحب أقل دين فى نهاية التقرير . اطبع فى نهاية التقرير إجمالى عدد الطلبة المدينين ، وإجمالى قيمة الدين، ومتوسط قيمة الدين.

١٢ – ٣٨ اكتب برنامجا ينتج تقرير الخطأ فى الأوامر ، وملف أوامر رئيسى .

• **المدخلات :**

| Field Name | Characteristics | Position |
|---|---|---|
| Customer Number | 9(6) | 1–6 |
| Customer Name | X(20) | 11–30 |
| Item Number | X(6) | 31–36 |
| Order Number | X(5) | 38–42 |
| Order Date | 9(6) | 44–49 |
| Order Quantity | 9(3) | 51–53 |
| Price | 999V99 | 55–59 |

البطاقات لا تكون مرتبة ، ويعرف رقم العميل ، ورقم الأمر ، ورقم العنصر بأنها حقول حرفية عددية .

• **المخرجات :**

١ – تقرير يحتوى على معلومات من هذه البطاقات تبين الأخطاء . وتأكد من صحة رقم العميل ، ورقم العنصر ، ورقم الأمر والتاريخ ، والكمية والسعر، ومن أن السعر أكبر من صفر .

٢ – ملف أوامر رئيسى على قرص ، يجب أن يكون مرتبا طبقا لرقم العميل ، مع الترتيب طبقا لرقم العنصر لكل عميل. ويوجد سجل واحد فى هذا الملف لكل عميل . الشكل التخطيطى لهذا الملف ، كذلك كما يلى :

| Field Name | Characteristics | Position |
|---|---|---|
| Customer Number | X(6) | 1–6 |
| Customer Name | X(20) | 7–26 |
| Number of Orders | 9 (COMP-3) | 27 |

يتبعه تاريخ لكل أمر :

| | |
|---|---|
| Item Number | X(6) |
| Order Number | X(5) |
| Date | 9(6) |
| Quantity | 9(3) |
| Price | 999V99 |

توجد أربعة أوامر على الأكثر للعميل الواحد ، وإجمالى طول السجل هو ١٢٧ بايت .

● **التشغيل** : التأكد من صحة سجلات المدخلات ، وطباعة تقرير بالبطاقات الخاطئة . يتم ترتيب البطاقات الصحيحة ؛ طبقاً لرقم العنصر داخل رقم العميل ، مع استخدام سجلات مرتبة فى إنتاج ملف رئيسى على قرص . إذا كانت هناك أكثر من أربعة سجلات للعميل الواحد .. فإنه لن يدخل إلا الأربعة سجلات الأولى فى الملف الرئيسى ، هذا بالإضافة إلى طباعة رسائل خطأ للأوامر الزائدة مع إهمال بطاقاتها .

١٢ – ٣٩ جدد بأوامر اليوم الملف الرئيسى الذى سبق إنتاجه فى التمرين السابق . وقد تريد استخدام خوارزمى خط الاتزان، الذى سبق التعرض له فى الفصل الحادى عشر .

● **المدخلات :**

(١) ملف أوامر رئيسى نفس التخطيط الموجود فى التمرين السابق ، ولكن به سجلات متغيرة الطول . بالسجل أى طول مطلوب لاحتوائه على الأوامر التى يشملها . ويبلغ طول السجلات التى تحتوى على أمر واحد 52 رمزا ، ويبلغ طول السجلات التى تحتوى على 4 أوامر 127 رمزا . هذا الملف هو ملف قرص .

٢ – ملف عمليات جارية للأوامر نفس التخطيط الموجود فى التمرين السابق ، وهو ملف بطاقات .

● **المخرجات :**

١ – ملف الأوامر الرئيسى المجدد .

٢ – مسجل عمليات جارية ، وتقرير مطبوع يبين التغيرات التى حدثت على الملف الرئيسى ، ويعرض رسائل الخطأ للسجلات التى لم تجدد فى الملف الرئيسى .

● **التشغيل :**

يرتب الملف الرئيسى طبقا لرقم العنصر داخل رقم العميل ، رتب ملف الأوامر ؛ بحيث يكون له نفس هذا الترتيب ( سبق التأكد من صحة كل بطاقات الأوامر ، وتحتوى على حقول عددية ) فى جزء التجديد .

إذا كان هناك أمر لعميل غير موجود فى الملف الرئيسى ، تطبع رسالة خطأ مع إهمال الأمر . إذا كان رقم العميل الموجود فى أحد الأوامر متفقاً مع نظيره فى الملف الرئيسى يحدث ما يلى :

إذا كان العنصر المطلوب موجوداً فعلا فى الملف الرئيسى .. فإن هذا العنصر يتجدد بإضافة كمية الأمر إلى الكمية الموجودة فى الملف الرئيسى ، ويتغير تاريخ الأمر فى الملف الرئيسى ليصبح تاريخ اليوم . أما إذا لم يوجد العنصر المطلوب فى الملف الرئيسى . تطبع رسالة خطأ إذا لم يكن هناك مكان لإضافة العنصر إلى الملف الرئيسى .

أما إذا كان هناك مكان لذلك .. فيضاف الى الملف الرئيسى ، ومع إدخاله .. يظل الترتيب كما هو طبقاً لرقم العنصر .

**تحذير :**

تأكد من كتابة كل السجلات الرئيسية التى تقرأ ، حتى إذا لم تكن هناك أوامر من العملاء .

# ملحق أ

# كلمات كوبل المحجوزة

# COBOL RESERVED WORDS

| الكلمة المحجوزة | كوبل 1974 النمطي | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطي* |
|---|---|---|---|---|
| ACCEPT | x | - | x | x |
| ACCESS | x | - | x | x |
| ACTUAL | - | - | x | - |
| ADD | x | - | x | x |
| ADVANCING | x | - | x | x |
| AFTER | x | - | x | x |
| ALL | x | - | x | x |
| ALPHABET | - | x | - | x |
| ALPHABETIC | x | - | x | x |
| ALPHANUMERIC | - | x | - | x |
| ALPHANUMERIC - EDITED | - | x | - | x |
| ALSO | x | - | x | x |
| ALTER | x | - | x | x |
| ALTERNATE | x | - | x | x |
| AND | x | - | x | x |
| ANY | - | x | - | x |
| APPLY | - | - | x | - |
| ARE | x | - | x | x |
| AREA | x | - | x | x |
| AREAS | x | - | x | x |
| ASCENDING | x | - | x | x |
| ASSIGN | x | - | x | x |
| AT | x | - | x | x |
| AUTHOR | x | - | x | x |

* إضافة من المترجم .

| الكلمة المحجوزة | كوبل 1974 النمطى | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطى* |
|---|---|---|---|---|
| BASIS | - | x | - | - |
| BEFORE | x | - | x | x |
| BEGINNING | - | - | x | - |
| BINARY | - | x | - | x |
| BIT | - | x | - | - |
| BITS | - | x | - | - |
| BLANK | x | - | x | x |
| BLOCK | x | - | x | x |
| BOOLEAN | - | x | - | - |
| BOTTOM | x | - | x | x |
| BY | x | - | x | x |
| CALL | x | - | x | x |
| CANCEL | x | - | x | x |
| CBL | - | - | x | - |
| CD | x | - | x | x |
| CF | x | - | x | x |
| CH | x | - | x | x |
| CHANGED | - | - | x | - |
| CHARACTER | x | - | x | x |
| CHARACTERS | x | - | x | x |
| CLASS* | - | - | - | x |
| CLOCK - UNITS | x | - | - | - |
| CLOSE | x | - | x | x |
| COBOL | x | - | - | - |
| CODE | x | - | x | x |
| CODE - SET | x | - | x | x |
| COLLATING | x | - | x | x |
| COLUMN | x | - | x | x |
| COMMA | x | - | x | x |
| COMMIT | - | x | - | - |
| COMMON | - | x | - | x |
| COMMUNICATION | x | - | x | x |
| COMP | x | - | x | x |
| COMP - 1 | - | - | x | - |
| COMP - 2 | - | - | x | - |
| COMP - 3 | - | - | x | - |
| COMP - 4 | - | - | x | - |
| COMPUTATIONAL | x | - | x | x |
| COMPUTATIONAL - 1 | - | - | x | - |
| COMPUTATIONAL - 2 | - | - | x | - |
| COMPUTATIONAL - 3 | - | - | x | - |
| COMPUTATIONAL - 4 | - | - | x | - |
| COMPUTE | x | - | x | x |

* إضافة من المترجم .

| الكلمة المحجوزة | كوبل 1974 النمطي | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطي* |
|---|---|---|---|---|
| CONFIGURATION | x | - | x | x |
| CONNECT | - | x | - | - |
| CONSOLE | - | - | x | - |
| CONTAINS | x | - | x | x |
| CONTENT | - | x | - | x |
| CONTINUE | - | x | - | x |
| CONTROL | x | - | x | x |
| CONTROLS | x | - | x | x |
| CONVERTING | - | x | - | x |
| COPY | x | - | x | x |
| CORE - INDEX | - | - | x | - |
| CORR | x | - | x | x |
| CORRESPONDING | x | - | x | x |
| COUNT | x | - | x | x |
| CSP | - | - | x | - |
| CURRENCY | x | - | x | x |
| CURRENT | - | x | - | - |
| CURRENT - DATE | - | - | x | - |
| C 01 | - | - | x | - |
| C 02 | - | - | x | - |
| C 03 | - | - | x | - |
| C 04 | - | - | x | - |
| C 05 | - | - | x | - |
| C 06 | - | - | x | - |
| C 07 | - | - | x | - |
| C 08 | - | - | x | - |
| C 09 | - | - | x | - |
| C 10 | - | - | x | - |
| C 11 | - | - | x | - |
| C 12 | - | - | x | - |
| DATA | x | - | x | x |
| DATE | x | - | x | x |
| DATE - COMPILED | x | - | x | x |
| DATE - WRITTEN | x | - | x | x |
| DAY | x | - | x | x |
| DAY - OF - WEEK | - | x | - | x |
| DB | - | x | - | - |
| DB - ACCESS - CONTROL - KEY | - | x | - | - |
| DB - DATA - NAME | - | x | - | - |
| DB - EXCEPTION | - | x | - | - |
| DB - RECORD - NAME | - | x | - | - |
| DB - SET - NAME | - | x | - | - |
| DB - STATUS | - | x | - | - |

* إضافة من المترجم

| الكلمة المحجوزة | كوبل 1974 النمطي | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطي* |
|---|---|---|---|---|
| DE | x | - | x | x |
| DEBUG | - | - | x | x |
| DEBUG - CONTENTS | x | - | x | x |
| DEBUG - ITEM | x | - | x | x |
| DEBUG - LINE | x | - | x | x |
| DEBUG - NAME | x | - | x | x |
| DEBUG - SUB - 1 | x | - | x | x |
| DEBUG - SUB - 2 | x | - | x | x |
| DEBUG - SUB - 3 | x | - | x | x |
| DEUGGING | x | - | x | x |
| DECIMAL - POINT | x | - | x | x |
| DECLARATIVES | x | - | x | x |
| DELETE | x | - | x | x |
| DELIMITED | x | - | x | x |
| DELIMTER | x | - | x | x |
| DEPENDING | x | - | x | x |
| DESCENDING | x | - | x | x |
| DESTINATION | x | - | x | x |
| DETAIL | x | - | x | x |
| DISABLE | x | - | x | - |
| DISCONNECT | - | - | - | - |
| DIS | - | - | x | x |
| DISPLAY | x | - | x | - |
| DISPLAY - n | - | x | - | - |
| DISPLAY - ST | - | - | x | x |
| DIVIDE | x | - | x | x |
| DIVISION | x | - | x | x |
| DOWN | x | - | x | x |
| DUPLICATE | - | x | - | - |
| DUPLICATES | x | - | x | x |
| DYNAMIC | x | - | x | x |
| EGI | x | - | x | x |
| EJECT | - | - | x | - |
| ELSE | x | - | x | x |
| EMI | x | - | x | x |
| EMPTY | - | x | - | - |
| ENABLE | x | - | x | x |
| END | x | - | x | x |
| END - ADD | - | x | - | x |
| END - CALL | - | x | - | x |
| END - COMPUTE | - | x | - | x |
| END - DELETE | - | x | - | x |
| END - DIVIDE | - | x | - | x |

* إضافة من المترجم .

| الكلمة المحجوزة | كوبل 1974 النمطي | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطي* |
|---|---|---|---|---|
| END - EVALUATE | - | x | - | x |
| END - IF | - | x | - | x |
| END - MULTIPLY | - | x | - | x |
| END - OF - PAGE | x | - | x | x |
| END - PERFORM | - | x | - | x |
| END - READ | - | x | - | x |
| END - RECEIVE | - | x | - | x |
| END - RETURN | - | x | - | x |
| END - REWRITE | - | x | - | x |
| END - SEARCH | - | x | - | x |
| END - START | - | x | - | x |
| END - STRING | - | x | - | x |
| END - SUBTRACT | - | x | - | x |
| END - UNSTRING | - | x | - | x |
| END - WRITE | - | x | - | - |
| ENDING | - | - | x | - |
| ENTER | x | - | x | - |
| ENTRY | - | - | x | x |
| ENVIRONMENT | x | - | x | x |
| EOP | x | - | x | x |
| EQUAL | x | - | x | - |
| EQUALS | - | x | - | - |
| ERASE | - | x | - | x |
| ERROR | x | - | x | x |
| ESI | x | - | x | x |
| EVALUATE | - | x | - | - |
| EVERY | x | - | x | - |
| EXAMINE | - | - | x | - |
| EXCEEDS | - | x | - | x |
| EXCEPTION | x | - | x | - |
| EXCLUSIVE | - | x | - | - |
| EXHIBIT | - | - | x | x |
| EXIT | x | - | x | - |
| EXOR | - | x | - | x |
| EXTEND | x | - | x | x |
| EXTERNAL | - | x | - | x |
| FALSE | - | x | - | x |
| FD | x | - | x | x |
| FILE | x | - | x | x |
| FILE - CONTROL | x | - | x | - |
| FILE - LIMIT | - | - | x | - |
| FILE - LIMITS | - | - | x | x |
| FILLER | x | - | x | x |

* إضافة من المترجم

| الكلمة المحجوزة | كوبل 1974 النمطي | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطي* |
|---|---|---|---|---|
| FINAL | x | - | x | - |
| FIND | - | x | - | - |
| FINISH | - | x | - | - |
| FIRST | x | - | - | x |
| FOOTING | x | - | x | x |
| FOR | x | - | x | - |
| FREE | - | x | - | x |
| FROM | x | - | x | - |
| FUNCTION | - | x | - | x |
| GENERATE | x | - | x | - |
| GET | - | x | - | x |
| GIVING | x | - | x | x |
| GLOBAL | - | x | - | x |
| GO | x | - | x | x |
| GREATER | x | - | x | x |
| GROUP | x | - | x | x |
| HEADING | x | - | x | x |
| HIGH - VALUE | x | - | x | x |
| HIGH - VALUES | x | - | x | x |
| I - O | x | - | x | x |
| I - O - CONTROL | x | - | x | - |
| ID | - | - | x | x |
| IDENTIFICATION | x | - | x | x |
| IF | x | - | x | x |
| IN | x | - | x | x |
| INDEX | x | - | x | - |
| INDEX -n | - | x | - | x |
| INDEXED | x | - | x | x |
| INDICATE | x | - | x | x |
| INITIAL | x | - | x | x |
| INTIALIZE | - | x | x | x |
| INITIATE | x | - | x | x |
| INPUT | x | - | x | x |
| INPUT - OUTPUT | x | - | x | - |
| INSERT | - | - | x | x |
| INSPECT | x | - | x | x |
| INSTALLATION | x | - | x | x |
| INTO | x | - | x | x |
| INVALID | x | - | x | x |
| IS | x | - | x | x |

* إضافة من المترجم

| الكلمة المحجوزة | كوبل 1974 النمطي | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطي* |
|---|---|---|---|---|
| JUST | x | - | x | x |
| JUSTIFIED | x | - | x | - |
| KEEP | - | x | - | x |
| KEY | x | - | x | x |
| LABEL | x | - | x | x |
| LAST | x | - | x | - |
| LD | - | x | - | x |
| LEADING | x | - | x | - |
| LEAVE | - | - | x | x |
| LEFT | x | - | x | x |
| LENGTH | x | - | x | x |
| LESS | x | - | x | x |
| LIMIT | x | - | x | x |
| LIMITS | x | - | x | x |
| LINAGE | x | - | x | x |
| LINAGE - COUNTER | x | - | x | x |
| LINE | x | - | x | x |
| LINE - COUNTER | x | - | x | x |
| LINES | x | - | x | x |
| LINKAGE | x | - | x | - |
| LOCALLY | - | x | - | x |
| LOCK | x | - | x | x |
| LOW - VALUE | x | - | x | x |
| LOW - VALUES | x | - | x | - |
| MEMBER | - | x | - | x |
| MEMORY | x | - | x | x |
| MERGE | x | - | x | x |
| MESSAGE | x | - | x | x |
| MODE | x | - | x | - |
| MODIFY | - | x | - | - |
| MODULES | x | - | x | x |
| MORE - LABELS | - | - | x | x |
| MOVE | x | - | x | x |
| MULTIPLE | x | - | x | - |
| MULTIPLY | x | - | x | x |
| NAMED | - | - | x | - |
| NATIVE | x | - | x | x |
| NEGATIVE | x | - | x | x |
| NEXT | x | - | x | - |
| NO | x | - | x | x |
| NOMINAL | - | x | x | - |

* إضافة من المترجم

| الكلمة المحجوزة | كوبل 1974 النمطى | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطى* |
|---|---|---|---|---|
| NOT | x | - | x | - |
| NOTE | - | - | x | x |
| NULL | - | x | - | x |
| NUMBER | x | - | x | x |
| NUMERIC | x | - | x | |
| NUMERIC - EDITED | - | x | - | x |
| OBJECT - COMPUTER | x | - | x | x |
| OCCURS | x | - | x | x |
| OF | x | - | x | x |
| OFF | x | - | x | x |
| OMITTED | x | - | x | x |
| ON | x | - | x | x |
| OPEN | x | - | x | x |
| OPTIONAL | x | - | x | x |
| OR | x | - | x | x |
| ORDER | - | x | - | x |
| ORGANIZATION | x | - | x | - |
| OTHER | - | x | - | x |
| OTHERWISE | - | - | x | x |
| OUTPUT | x | - | x | - |
| OVERFLOW | x | - | x | x |
| OWNER | - | x | - | x |
| PACKED - DECIMAL | - | x | - | x |
| PADDING | - | x | - | x |
| PAGE | x | - | x | x |
| PAGE - COUNTER | x | - | x | - |
| PASSWORD | - | - | x | x |
| PERFORM | x | - | x | x |
| PF | x | - | x | x |
| PH | x | - | x | x |
| PIC | x | - | x | x |
| PICTURE | x | - | x | x |
| PLUS | x | - | x | x |
| POINTER | x | - | x | x |
| POSITION | x | - | x | - |
| POSITIONING | - | - | x | x |
| POSITIVE | x | - | x | x |
| PRINTING | x | - | - | - |
| PRIOR | - | x | - | x |
| PROCEDURE | x | - | x | x |
| PROCEDURES | x | - | x | x |
| PROCEED | x | - | x | - |

* إضافة من المترجم

| الكلمة المحجوزة | كوبل 1974 النمطى | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطى* |
|---|---|---|---|---|
| PROCESSING | - | - | x | x |
| PROGRAM | x | - | x | x |
| PROGRAM - ID | x | - | x | x |
| PROTECTED | - | x | - | x |
| PURGE | - | x | - | x |
| QUEUE | x | - | x | x |
| QUOTE | x | - | x | x |
| QUOTES | x | - | x | x |
| RANDOM | x | - | x | - |
| RD | x | - | x | - |
| READ | x | - | x | - |
| READY | - | - | x | x |
| REALM | - | x | - | - |
| REALMS | - | x | - | x |
| RECEIVE | x | - | x | - |
| RECONNECT | - | x | - | - |
| RECORD | x | - | x | x |
| RECORD - NAME | - | x | - | x |
| RECORD - OVERFLOW | - | - | x | x |
| RECORDS | x | - | x | x |
| REDEFINES | x | - | x | x |
| REEL | x | - | x | x |
| REFERENCE | - | x | - | x |
| REFERENCES | x | - | x | - |
| RELATIVE | x | - | x | x |
| RELEASE | x | - | x | - |
| RELOAD | - | - | x | x |
| REMAINDER | x | - | x | x |
| REMARKS | - | - | x | - |
| REMOVAL | x | - | x | - |
| RENAMES | x | - | x | x |
| REORG - CRITERIA | - | - | x | x |
| REPEATED | - | x | - | x |
| REPLACE | - | x | - | x |
| REPLACING | x | - | x | x |
| REPORT | x | - | x | - |
| REPORTING | x | - | x | - |
| REPORTS | x | - | x | x |
| REREAD | - | - | x | x |
| RERUN | x | - | x | - |
| RESERVE | x | - | x | - |
| RESET | x | - | x | x |
| RETAINING | - | x | - | - |

* إضافة من المترجم

| الكلمة المحجوزة | كوبل 1974 النمطي | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطي* |
|---|---|---|---|---|
| RETRIEVAL | - | x | - | - |
| RETURN | x | - | x | x |
| RETURN - CODE | - | - | x | x |
| REVERSED | x | - | x | x |
| REWIND | x | - | x | x |
| REWRITE | x | - | x | x |
| RF | x | - | x | - |
| RH | x | - | x | x |
| RIGHT | x | - | x | x |
| ROLLBACK | - | x | - | x |
| ROUNDED | - | - | x | x |
| RUN | x | - | x | x |
| SAME | x | - | x | x |
| SD | x | - | x | x |
| SEARCH | x | - | x | - |
| SECTION | x | - | x | x |
| SECURITY | x | - | x | x |
| SEEK | - | - | x | x |
| SEGMENT | x | - | x | - |
| SEGMENT - LIMIT | x | - | x | x |
| SELECT | x | - | x | x |
| SELECTIVE | - | - | x | x |
| SEND | x | - | x | x |
| SENTENCE | x | - | x | x |
| SEPARATE | x | - | x | x |
| SEQUENCE | x | - | x | - |
| SEQUENTIAL | x | - | x | x |
| SET | x | - | x | x |
| SETS | - | x | - | - |
| SIGN | x | - | x | - |
| SIZE | x | - | x | - |
| SKIP-1 | - | - | x | x |
| SKIP-2 | - | - | x | - |
| SKIP-3 | - | - | x | - |
| SORT | x | - | x | x |
| SORT-CODE - SIZE | - | - | x | - |
| SORT-FILE - SIZE | - | - | x | - |
| SORT-MERGE | x | - | x | - |
| SORT-MESSAGE | - | - | x | x |
| SORT-MODE - SIZE | - | - | x | x |
| SORT-RETURN | - | - | x | x |
| SOURCE | x | - | x | x |
| SOURCE - COMPUTER | x | - | x | x |

* إضافة من المترجم .

| الكلمة المحجوزة | كوبل 1974 النمطي | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطي* |
|---|---|---|---|---|
| SPACE | x | - | x | x |
| SPACES | x | - | x | x |
| SPECIAL - NAMES | x | - | x | x |
| STANDARD | x | - | x | x |
| STANDARD-1 | x | - | x | x |
| STANDARD-2 | - | x | - | x |
| START | x | - | x | - |
| STATUS | x | - | x | x |
| STOP | x | - | x | x |
| STORE | - | - | x | x |
| STRING | x | - | x | x |
| SUB-QUEUE- -1 | x | - | x | - |
| SUB-QUEUE-2 | x | - | x | x |
| SUB- QUEUE-3 | x | - | x | x |
| SUB - SCHEMA | - | x | - | x |
| SUBTRACT | x | - | x | x |
| SUM | x | - | x | x |
| SUPPRESS | x | - | x | x |
| SYMBOLIC | x | - | x | - |
| SYNC | x | - | x | - |
| SYNCHRONIZED | x | - | x | - |
| SYSIN | - | - | x | - |
| SYSOUT | - | - | x | - |
| SYSPUNCH | - | - | x | x |
| S 01 | - | - | x | - |
| S 0 2 | - | - | x | |
| | | | | x |
| TABLE | x | - | x | - |
| TALLY | - | - | x | x |
| TALLYING | x | - | x | x |
| TAPE | x | - | x | x |
| TENANT | - | x | - | x |
| TERMINAL | x | - | x | x |
| TERMINATE | x | - | x | x |
| TEST | - | x | - | x |
| TEXT | x | - | x | x |
| THAN | x | - | x | x |
| THEN | - | - | x | - |
| THROUGH | x | - | x | x |
| THRU | x | - | x | x |
| TIME | x | - | x | - |
| TIME - OF - DAY | - | - | x | x |
| TIMES | x | - | x | x |
| TO | x | - | x | x |

* إضافة من المترجم .

| الكلمة المحجوزة | كوبل 1974 النمطي | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطي* |
|---|---|---|---|---|
| TOP | x | - | x | - |
| TOTALED | - | - | x | - |
| TOTALING | - | - | x | - |
| TRACE | - | - | x | x |
| TRACK - AREA | - | - | x | - |
| TRACK - LIMIT | - | - | x | x |
| TRACKS | - | - | x | x |
| TRAILING | x | - | x | x |
| TRANSFORM | - | - | x | - |
| TRUE | - | x | - | x |
| TYPE | x | - | x | - |
| UNEQUAL | - | x | - | x |
| UNIT | x | - | x | x |
| UNSTRING | x | - | x | - |
| UNTIL | x | - | x | x |
| UP | x | - | x | - |
| UPDATE | - | x | - | - |
| UPON | x | - | x | - |
| UPSI - 0 | - | - | x | - |
| UPSI - 1 | - | - | x | - |
| UPSI - 2 | - | - | x | - |
| UPSI - 3 | - | - | x | - |
| UPSI - 4 | - | - | x | - |
| UPSI - 5 | - | - | x | x |
| UPSI - 6 | - | - | x | - |
| UPSI - 7 | - | - | x | x |
| USAGE | x | - | x | x |
| USAGE - MODE | - | x | - | x |
| USE | x | - | x | x |
| USING | x | - | x | x |
| VALUE | x | - | x | x |
| VALUES | x | - | x | x |
| VARYING | x | - | x | x |
| WHEN | x | - | x | - |
| WHEN - COMPILED | - | - | x | x |
| WITH | x | - | x | - |
| WITHIN | - | x | - | x |
| WORDS | x | - | x | x |
| WORKING - STORAGE | x | - | x | x |
| WRITE | x | - | x | - |
| WRITE - ONLY | - | - | x | x |

* إضافة من المترجم .

| الكلمة المحجوزة | كوبل 1974 النمطي* | كوبل CODASYL 80 | كوبل IBM OS/VS | كوبل 1985 النمطي* |
|---|---|---|---|---|
| ZERO | x | - | x | x |
| ZEROES | x | - | x | x |
| ZEROS | x | - | x | x |
| ALPHABETIC - LOWR * | - | - | - | x |
| ALPHABETIC - UPPER * | - | - | - | x |
| + | x | - | x | x |
| - | x | - | x | x |
| * | x | - | x | x |
| / | x | - | x | x |
| ** | x | - | x | x |
| < | x | - | x | x |
| > | x | - | x | x |
| = | x | - | x | x |
| >=* | - | - | - | x |
| <=* | - | - | - | x |

* إضافة من المترجم .

# ملحق ب

# تسلسل التتابع

# Collating Sequences

الترتيب التصاعدي لنظام EBCDIC

| المعنـى | الـرمز | تشكيل البت | التمثيل العشـرى |
|---|---|---|---|
| – علامة السنت | ¢ | 01001010 | 74 |
| – نقطة أو علامة عشرية | . | 01001011 | 75 |
| – إشارة أقل من | < | 01001100 | 76 |
| – قوس أيسر | ( | 01001101 | 77 |
| – اشارة موجب | + | 01001110 | 78 |
| – عمود رأسى ، أو منطقية | \| | 01001111 | 79 |
| – علامة إضافة | & | 01010000 | 80 |
| – علامة تعجب | ! | 01011010 | 90 |
| – علامة دولار | $ | 01011011 | 91 |
| – نجمة | * | 01011100 | 92 |
| – قوس أيمن | ) | 01011101 | 93 |
| – فاصلة منقوطة | ; | 01011110 | 94 |
| – NOT منطقية | ¬ | 01011111 | 95 |
| – إشارة سالب أو شرطة | – | 01100000 | 96 |
| – شرطة مائلة | / | 01100001 | 97 |
| – فاصلة | , | 01101011 | 107 |

| المعنى | الرمز | تشكيل البت | التمثيل العشرى |
|---|---|---|---|
| – علامة النسبة المئوية | % | 01101100 | 108 |
| – شرطة تحت الحرف | - | 01101101 | 109 |
| – إشارة أكبر من | > | 01101110 | 110 |
| – علامة استفهام | ? | 01101111 | 111 |
| – نقطتان علويتان | : | 01111100 | 122 |
| – إشارة الرقم | # | 01111011 | 123 |
| – اشارة عند | @ | 01111010 | 124 |
| – علامة تنصيص فردية | ' | 01111101 | 125 |
| – علامة تساوى | = | 01111110 | 126 |
| – علامة تنصيص مزدوجة | " | 01111111 | 127 |
| | a | 10000001 | 129 |
| | b | 10000010 | 130 |
| | c | 10000011 | 131 |
| | d | 10000100 | 132 |
| | e | 10000101 | 133 |
| | f | 10000110 | 134 |
| | g | 10000111 | 135 |
| | h | 10001000 | 136 |
| | i | 10001001 | 137 |
| | j | 10010001 | 145 |
| | k | 10010010 | 146 |
| | l | 10010011 | 147 |
| | m | 10010100 | 148 |
| | n | 10010101 | 149 |
| | o | 10010110 | 150 |
| | p | 10010111 | 151 |
| | q | 10011000 | 152 |

| التمثيل العشــــرى | تشكيل البت | الـرمز | المعنـــى |
|---|---|---|---|
| 153 | 10011001 | r | |
| 162 | 10100010 | s | |
| 163 | 10100011 | t | |
| 164 | 10100100 | u | |
| 165 | 10100101 | v | |
| 166 | 10100110 | w | |
| 167 | 10100111 | x | |
| 168 | 10101000 | y | |
| 169 | 10101001 | z | |
| 193 | 11000001 | A | |
| 194 | 11000010 | B | |
| 195 | 11000011 | C | |
| 196 | 11000100 | D | |
| 197 | 11000101 | E | |
| 198 | 11000110 | F | |
| 199 | 11000111 | G | |
| 200 | 11001000 | H | |
| 201 | 11001001 | I | |
| 209 | 11010001 | J | |
| 210 | 11010010 | K | |
| 211 | 11010011 | L | |
| 212 | 11010100 | M | |
| 213 | 11010101 | N | |
| 214 | 11010110 | O | |
| 215 | 11010111 | P | |
| 216 | 11011000 | Q | |

| المعنى | الرمز | تشكيل البت | التمثيل العشرى |
|---|---|---|---|
| | R | 11011001 | 217 |
| | S | 11100010 | 226 |
| | T | 11100011 | 227 |
| | U | 11100100 | 228 |
| | V | 11100101 | 229 |
| | W | 11100110 | 230 |
| | X | 11100111 | 231 |
| | Y | 11101000 | 232 |
| | Z | 11101001 | 233 |
| | 0 | 11110000 | 240 |
| | 1 | 11110001 | 241 |
| | 2 | 11110010 | 242 |
| | 3 | 11110011 | 243 |
| | 4 | 11110100 | 244 |
| | 5 | 11110101 | 245 |
| | 6 | 11110110 | 246 |
| | 7 | 11110111 | 247 |
| | 8 | 11111000 | 248 |
| | 9 | 11111001 | 249 |

الترتيب التصاعدى لنظام ASCII

| المعنـــى | الـرمز | تشكيل البت | التمثيل العشـــرى |
|---|---|---|---|
| – لا شيء (صفر) | | 00000000 | 0 |
| – فراغ | SP | 00100000 | 32 |
| – OR منطقية | ! | 00100001 | 33 |
| – علامة تنصيص مزدوجة | " | 00100010 | 34 |
| – إشارة رقم | # | 00100011 | 35 |
| – علامة دولار | $ | 00100100 | 36 |
| – نسبة مئوية | % | 00100101 | 37 |
| – علامة إضافة | & | 00100110 | 38 |
| – علامة تنصيص فردية | ' | 00100111 | 39 |
| – قوس مفتوح | ( | 00101000 | 40 |
| – قوس مغلق | ) | 00101001 | 41 |
| – نجمة | * | 00101010 | 42 |
| – زائد | + | 00101011 | 43 |
| – فاصلة | , | 00101100 | 44 |
| – ناقص أو شرطة | - | 00101101 | 45 |
| – نقطة أو علامة عشرية | . | 00101110 | 46 |
| – شرطة مائلة للخلف | / | 00101111 | 47 |
| | 0 | 00110000 | 48 |
| | 1 | 00110001 | 49 |
| | 2 | 00110010 | 50 |
| | 3 | 00110011 | 51 |
| | 4 | 00110100 | 52 |
| | 5 | 00110101 | 53 |
| | 6 | 00110110 | 54 |
| | 7 | 00110111 | 55 |
| | 8 | 00111000 | 56 |
| | 9 | 00111001 | 57 |

| المعنى | الرمز | تشكيل البت | التمثيل العشرى |
|---|---|---|---|
| – نقطتان علويتان | : | 00111010 | 58 |
| – فاصلة منقوطة | ; | 00111011 | 59 |
| – أقل من | < | 00111100 | 60 |
| –يساوى | = | 00111101 | 61 |
| – أكبر من | > | 00111110 | 62 |
| – علامة استفهام | ? | 00111111 | 63 |
| – اشارة عند | @ | 01000000 | 64 |
| | A | 01000001 | 65 |
| | B | 01000010 | 66 |
| | C | 01000011 | 67 |
| | D | 01000100 | 68 |
| | E | 01000101 | 69 |
| | F | 01000110 | 70 |
| | G | 01000111 | 71 |
| | H | 01001000 | 72 |
| | I | 01001001 | 73 |
| | J | 01001010 | 74 |
| | K | 01001011 | 75 |
| | L | 01001100 | 76 |
| | M | 01001101 | 77 |
| | N | 01001110 | 78 |
| | O | 01001111 | 79 |
| | P | 01010000 | 80 |
| | Q | 01010001 | 81 |
| | R | 01010010 | 82 |
| | S | 01010011 | 83 |
| | T | 01010100 | 84 |

| التمثيل العشـــرى | تشكيل البت | الـرمز | المعنـــى |
|---|---|---|---|
| 85 | 01010101 | U | |
| 86 | 01010110 | V | |
| 87 | 01010111 | W | |
| 88 | 01011000 | X | |
| 89 | 01011001 | Y | |
| 90 | 01011010 | Z | |
| 91 | 01011011 | [ | قوس مربع مفتوح |
| 92 | 01011100 | \ | – شرطة مائلة عكسية |
| 93 | 01011101 | ] | – قوس مربع مغلق |
| 94 | 01011110 | ^ | – منطقية |
| 95 | 01011111 | _ | – شرطة تحت الحرف |
| 96 | 01100000 | ` | grave accent |
| 97 | 01100001 | a | |
| 98 | 01100010 | b | |
| 99 | 01100011 | c | |
| 100 | 01100100 | d | |
| 101 | 01100101 | e | |
| 102 | 01100110 | f | |
| 103 | 01100111 | g | |
| 104 | 01101000 | h | |
| 105 | 01101001 | i | |
| 106 | 01101010 | j | |
| 107 | 01101011 | k | |
| 108 | 01101100 | l | |
| 109 | 01101101 | m | |

| المعنـــى | الـرمز | تشكيل البت | التمثيل العشـــرى |
|---|---|---|---|
| | n | 01101110 | 110 |
| | o | 01101111 | 111 |
| | p | 01110000 | 112 |
| | q | 01110001 | 113 |
| | r | 01110010 | 114 |
| | s | 01110011 | 115 |
| | t | 01110100 | 116 |
| | u | 01110101 | 117 |
| | v | 01110110 | 118 |
| | w | 01110111 | 119 |
| | x | 01111000 | 120 |
| | y | 01111001 | 121 |
| | z | 01111010 | 122 |
| – قوس {مفتوح | { | 01111011 | 123 |
| – خط علوى | ¦ | 01111100 | 124 |
| – قوس مغلق} | } | 01111101 | 125 |
| tilde | ~ | 01111110 | 126 |

# ملحق جـ

# اعتبارات كوبل الثمانينيات

# Cobol' 80 Considerations

تعمل لجان CODASYL , ANSI حديثا ( طبقا لما جاء فى الكتاب ) على صيغة جديدة لكوبل الثمانينيات الميلادية، وكأول محاولة أذيعت ، عام 1981 م ، هى كوبل 80 النمطى ، الذى يشمل تغييرات تجعل بعض المبرمجين بصيغة كوبل 1974 النمطية غير متوافقين مع كوبل 80 . وحاليا تجرى مراجعة أكثر للنمطية الجديدة ، مع وجود نقد بناء وحوار حولها .

وفيما يلى بعض التغييرات الأكثر أهمية التى يمكن أن يتفق عليها :

١ – القيد على استخدام 3 دلائل ( أو فهارس ) ، و 3 مستويات لأجزاء OCCURS متداخلة أزيل ( الى 48 ) .

٢ – يمكن تعريف السلاسل الجزئية لعناصر البيانات فى عبارات جزء الإجراءات : يحدد 7 بايت للعنصر DATA - ITEM الذى يبدأ بالبايت الثالث .

٣ – يمكن أن تكون للعنصر DEPENDING ON فى جزء OCCURS ... DEPENDING ON القيمة صفر (محددا محتويات صفرية للجدول) .

٤ – يمكن لعنصر البيانات الذى يعرف عنصر بيانات آخر أن يكون أقصر منه.

٥ – تنتج ACCEPT ... FROM DAY - OF - WEEK رقما مناسبا يقع بين 1 و 7 .

٦ – يسمح الآن بكلمة TO مع GIVING فى عملية الجمع .

ADD ITEM - A TO ITEM - B GIVING ITEM - C.

٧ – يمكن استخدام عبارة MOVE لنقل عنصر منقح إلى عنصر عددى غير منقح ، ولا يسري التنقيح أثناء العملية .

٨ – يسمح الآن بسجلات متغيرة الطول مع WRITE ... FROM, RETURN ... INTO, READ ... INTO, RELEASE ... FROM .

٩ – يتغير وصف محتويات المسجل الخاص DEBUG - ITEM ؛ تبعاً لتاثير ذلك على USE FOR DEBUGGING DECLARATIVES.

١٠ – عندما يكون عنصر DEPENDING ON جزءا من سجل متغير الطول ، ويكون حقلا مستقبلا ، تستخدم أقصى قيمة ممكنة لعنصر DEPENDING ON ( بدلا من قيمته الحالية ) فى تحديد طول الحقل المستقبل .

١١ – هناك شرطا فئة CLASS جديدان ، هما :

ALPHABETIC - LOWER, ALPHABETIC - UPPER . يغطى شرط ALPHABETIC الحالى كلاً من الحروف العليا ( الحروف الكبيرة ) ، والحروف الدنيا ( الحروف الصغيرة ) .

١٢ – حذفت عبارة ALTER من اللغة .

١٣ – ينتهى تقويم التعبيرات الشرطية المركبة بمجرد تحديد القيمة الحقيقية النهائية للتعبير . وعلى هذا .. إذا كان A خطأ فى "IF A AND B AND C..." .. فلن يقوم B أو C ؛ لأنه أصبح معروفا على الفور أن التعبير خطأ.

١٤ – اتسع تعريف PERFORM للسماح بتحديد العبارات التى تنفذ فى سطر على التوالى :

```
PERFORM UNTIL FLAG-IS-SET
    MOVE EMPLOYEE-NAME TO DETAIL-LINE-NAME
    ADD 1 TO NUMBER-PRINTED
    WRITE DETAIL-LINE . . .
    PERFORM SOME-CALCULATION
    ADD EMPLOYEE-AMOUNT TO TOTAL-AMOUNT
END-PERFORM
```

قواعد تنفيذ العبارات المحددة هى نفسها كما لو كانت العبارات موجودة بمقطع مستقل بها ( وليكن PARA - A مثلا ) وكان يستخدم مايلى:

```
PERFORM PARA-A UNTIL FLAG-IS-SET
```

لاحظ أنه عندما تحدد العبارات المراد تنفيذها فى سطر على التوالى: (١) لا يوضع اسم مقطع أو قسم فى عبارة -PER FORM (٢) ويكون مطلوب عبارة جديد END - PERFORM ؛ لتحديد نهاية مجموعة العناصر المشمولة فى عبارة -PER FORM .

١٥ – اتسعت PERFORM ... UNTIL لتسمح بمواصفات تبين ضرورة اختبار شرط UNTIL قبل أو بعد تنفيذ العبارات أو المقاطع المحددة :

```
PERFORM SAMPLE-PARAGRAPH
    WITH TEST AFTER
    UNTIL WE-ARE-DONE
```

أو

```
PERFORM WITH TEST BEFORE
    UNTIL TEST-IS-SATISFIED
        ADD...
        MOVE...
        READ...
END-PERFORM
```

باستخدام TEST - AFTER .. تنفذ العبارات المحددة مرة واحدة على الأقل، حتى إذا كان الشرط متحققا فى البداية (DO UNTIL) ، و باستخدام TEST - BEFORE (الحالة التقليدية) .. لا تنفذ العبارات المحددة عندما يكون الشرط متحققا فى البداية .

١٦ – تنتهى عبارة IF بعبارة جديدة هى END - IF بدلا من نهايتها باستخدام النقطة .

```
IF A EQUAL B
    ADD...
    MOVE...
END-IF
```

أو

```
IF X NOT EQUAL Y
    MOVE...
    PERFORM...
ELSE
    ADD...
    READ...
END-IF
```

١٧ – أضيفت عبارة جديدة EVALUATE إلى الكوبل لتنفيذ هيكل الحالة . ونوضح ذلك بإعادة كتابة مثال (٧ – ٢٦) على النحو التالى:

```
EVALUATE TRANSACTION-CODE
    WHEN 1 PERFORM CREATE-NEW-MASTER-RECORD
    WHEN 2 PERFORM DELETE-MASTER-RECORD
    WHEN 3 PERFORM CHANGE-EXISTING-MASTER-RECORD
    WHEN OTHER
        PERFORM INVALID-TRANSACTION-CODE-ROUTINE
END-EVALUATE
```

لاحظ جزء اصطياد الخطأ WHEN OTHER ، والإنهاء الصريح END- EVALUATE .

## كلمة من المترجم عن الاختلافات الرئيسية بين كوبل 1985 النمطى ،وكوبل 1974 النمطى

لقد ظهرت فعلا صيغة الكوبل القياسية المعروفة بكوبل ١٩٨٥ النمطى ، وفيما يلى الاختلافات الرئيسية بينها وبين كوبل ١٩٧٤ النمطى ، هذا .. مع ملاحظة ملحق أ ، وأن هناك كلمات محجوزة أضيفت إلى صيغة كوبل ١٩٨٥ النمطية ، مع حذف بعض كلمات من صيغة كوبل ١٩٧٤ النمطية منها .

١ - أصبح من الممكن استخدام كلمتى GIVING , TO فى نفس عبارة الجمع .

٢ - أصبح فى الإمكان استخدام مؤشرات لإنهاء العمليات الحسابية ، وهى:

END - ADD
END - SUBTRACT
END - DEVIDE
END - MULTIPLY
END - COMPUTE

كما أصبح فى الإمكان استخدام مؤشرات لإنهاء عبارات أخرى على النحو التالى :

END - IF
END - EVALUATE
END - PERFORM
END - READ
END - SEARCH
END - WRITE

واستخدام هذه المؤشرات يجعل العبارات الشرطية أكثر ترتيبا مما يسهل من البرمجة المرتبة .

٣ - أصبح من الممكن استخدام كلمة THEN عند كتابة الشروط ؛ مما يجعل الكوبل متفقا مع هيكل IF - THEN - ELSE بالنسبة لهيكل الاختيار المرتب .

٤ - أصبح من الممكن استخدام PERFORM ، وكتابة العبارات المراد تنفيذها فى سطر على التوالى. (الملاحظة رقم ١٤ أعلاه)

٥ - أصبح بالإمكان استخدام جزء TEST AFTER فى عبارة PERFORM .

٦ - أصبح من الممكن وضع قيم ابتدائية لمجموعة عناصر باستخدام فعل INITIALIZE .

٧ - أصبح فى الإمكان استخدام عدد من المستويات يصل الى 7 مع OCCURS .

٨ - يمكن لعنصر OCCURS أن يحتوى على محتويات ابتدائية باستخدام جزء VALUE و هذا يلغى الحاجة الى REDE-FINE الجدول أو المنظومة .

٩ - ادخل فعلاً جديداً EVALUATE وذلك لتنفيذ هيكل الحالة بالكوبل .

١٠ - أصبح فى الإمكان نقل عناصر منقحة إلى عناصر عددية .

١١ - أصبح فى الإمكان استخدام كلمة الكوبل المحجوزة DAY - OF - WEEK التى تشغل محتوياتها خانة واحدة ؛ للدلالة على رقم اليوم فى الأسبوع والذى يتراوح من 1 الى 7 .

١٢ - أصبح النقل النسبى ممكنا من حقل لآخر ؛ فيمكن نقل جزء من حقل كبير إلى حقل صغير .

١٣ - أصبح من الممكن للثوابت غير العددية أن تشغل حتى 160 رمز ، بينما كان الحد الأعلى فى كوبل ١٩٧٤ النمطى هو 120 رمز فقط .

١٤ - لا تحتاج عبارة EXIT إلى إن تكون هى العبارة الوحيدة فى المقطع الذى يحتويها ، بينما - من الممنوع فى كوبل ١٩٧٤ النمطى كتابة أى شىء فى المقطع المكتوب فيه EXIT باستثنائها هى بالطبع .

١٥ – يمكن أن تكون أسماء الإجراءات المستخدمة مع عبارة SORT أسماء مقاطع أو اسماء أقسام ، بينما يشترط فيها أن تكون أسماء أقسام فى كوبل ١٩٧٤ النمطى.

١٦ – أصبح قسم التشكيل اختياريا . فى الواقع .. أصبح جزء التشكيل اختياريا.

١٧ – أصبحت RECORD CONTAINS, BLOCK CONTAINS اختيارية .

١٨ – أضيف المؤثران الحسابيان

( > = ) IS GREATER THAN OR EQUAL , ( < = ) IS LESS THAN OR EQUAL ,

١٩ – أصبح من الممكن استخدام نفى عند حدوث خطأ فى الحجم ، ونهاية الملف ، والمفتاح غير الصحيح ؛ أى أصبح مسموحا باستخدام ، و NOT AT END , NOT ON SIZE ERROR , NOT INVALID KEY .

٢٠ – أضيف الجزء WITH NO ADVANCING إلى عبارة DISPLAY ؛ لجعل التداخل بين مشغل الكمبيوتر والكمبيوتر أكثر صداقة .

٢١ – أصبح من الممكن استخدام الحروف الأبجدية الصغيرة فى الثوابت الحرفية عددية . وتعتبر كأنها حرفية ، وتجتاز اختبار ALPHABETIC .

# المصطلحات العلمية ( إنجليزى عربى )

## A

| | |
|---|---|
| A (picture character) | A (رمز صورة) |
| A margin | المنطقة |
| Abbreviaation | أختصار |
| Abbreviation of compound conditions | اختصار شروط مركبة |
| ABEND | نهاية غير طبيعية |
| Absolute value | قيمة مطلقة |
| ACCEPT | عبارة اقبل |
| Access arm | ذراع اتصال |
| ACCESS IS | الاتصال يكون |
| Access modes | حالات الاتصال |
| Access time | وقت الاتصال |
| Accumulators | مركمات |
| Action Stub | جزء أجزاء وهى |
| ADD | تجمع |
| Adding an entry | إضافة محتوى |
| Address | عنوان |
| Address calculations: | حسابات عنوان |
| AFTER | بعد |
| Addvancing mnemonic - name | مع تقديم اسم خاص |
| ADVANCING PAGE | مع تقديم صفحة |
| AFTER ADVANCING | بعد تقديم |
| Algorithm | خوارزمى |
| ALL literal | كل ثوابت |
| ALPHABETIC | ابجدى |

| | |
|---|---|
| Alphabetic data | بيانات حرفية |
| Alphameric (alphanumeric) data | بيانات حرفية عددية (غير عددية) |
| ALU (arithmetic-logic unit) | وحدة حسابات ومنطق |
| AND | و |
| Applications program | برنامج تطبيق |
| Arithmetic, efficienay of | حساب ، كفاءة |
| Arithmetic- operators | مؤثرات حسابية |
| Arrays, | منظومات |
| ASCENDING KEY | مفتاح تصاعدى |
| ASCENDING/DESCENDING KEY | مفتاح تصاعدى أو تنازلى |
| Ascending sort | ترتيب تصاعدى |
| ASCII | الشفرة الأمريكية القياسية لتبادل المعلومات |
| ASSIGN TO | محدد له |
| Assumed | مفترضة |
| Assumed decimal point | علامة عشرية مفترضة |
| Asterisk (editing character) | نجمة ( رمز تنقيح ) |
| AT END | عند النهاية |
| AT END-OF-PAGE | عند نهاية الصفحة |
| AT EOP | عند نهاية الصفحة |
| AUTHOR | مؤلف |
| Automatic page overflow | سريان زائد تلقائى للصفحة |
| Auxiliary storage | تخزين مساعد |

# B

| | |
|---|---|
| B (picture character) | B ( رمز صورة ) |
| B margin | المنطقة ب |
| Backup | احتياطى |
| Balanced-line sequential file update | تجديد ملف توازن تتابعى |
| Batch processing | تشغيل دفعات |
| Binary digit, | رقم ثنائى |
| Binary operators, | مؤثرات ثنائية |
| Binary search, | بحث ثنائى |
| Binary-twos-complement data | بيانات مكملة ازواج ثنائية |
| Bit | بت |

| | |
|---|---|
| Blank lines, | أسطر فارغة |
| BLANK WHEN ZERO | فراغات بدلا من الأصفار |
| Block: | مجموعة |
| BLOCK CONTAINS, | المجموعة تحتوى |
| BLOCK CONTAINS 0 CHARACTERS | المجموعة تحتوى صفرا من الرموز |
| Block descriptor word | كلمة واصف مجموعة |
| Blocking | تجميع |
| Blocking factor | معامل التجميع |
| Bottom margin | الهامش السفلى |
| Braces, | أقواس |
| Brackets | أقواس |
| Branch instruction | أمر تفريغ |
| Bubble | فقاعة |
| Bubble Sort | ترتيب الفقاعة |
| Buffer | ذكرة احتياطية |
| Byte | بايت |

# C

| | |
|---|---|
| Cslculating Sort record fields | حساب حقول سجل ترتيب |
| Card punch | مثقب بطاقات |
| Card reader | قارىء بطاقات |
| Carriage control C01,C02 | تحكم العربة |
| Case structure | هيكل الحالة |
| COBOL | كوبل |
| CHANGED | تغيير |
| CHANGED NAMES | أسماء متغيرة |
| Check protection | حماية الشيكات |
| Class condition | شرط فئة |
| CLOSE | أغلق |
| COBOL | كوبل |
| Coding form | صيغة كتابة شفرة |
| Coding - testing plan | خطة أختبار الشفرة |
| Coding standards | نمطيات كتابة شفرة |
| Cohesion of modules | تماسك الأجزاء |
| Construction Of sort records | اعداد سجلات ترتيب |

| | |
|---|---|
| Collating sequence, | تسلسل التتابع |
| COLLATING SEQUENCE IS | تسلسل التتابع هو |
| Column | عمود |
| Comma (editing character) | فاصلة ( رمز تنقيح ) |
| Command language | لغة أوامر |
| Commands | أوامر |
| Comments | تعليقات |
| Completion of documetation | اتمام التوثيق |
| COMP | حسابى |
| COMP - 3 | حسابى - ٣ |
| Compared to subscript | مقارنة بدليل |
| COMP data | بيانات مضغوطه |
| COMP - 3 data | بيانات مضغوطه - ٣ |
| COMPUTATIONAL | حسابى |
| COMPUTATIONAL - 3 | حسابى - ٣ |
| Comparison: | مقارنه |
| Compilation (diagram) | ترجمة ( رسم ) |
| Compiler | مترجم |
| Compound conditions | شروط مركبة |
| COMPUTE | احسب |
| Computer system | نظام كمبيوتر |
| Condition, | شرط |
| Condition-names | أسماء شرطية |
| Condition stub | جزء شرطى وهمى |
| Conditional error | خطأ شرطى |
| Conditional statements | عبارات شرطية |
| CONFIGURATION SECTION | قسم التشغيل |
| Connector symbol | رمز الواصل |
| CONSOLE | شاشة مرئية |
| Console terminal | نهاية طرفية بها شاشة |
| Continuation rules | قواعد مستمرة |
| Control break | تحكم متقطع |
| Control cards | بطاقات تحكم |
| Control fields | حقول تحكم |
| Control totals | إجماليات تحكم |

| | |
|---|---|
| Control unit (CU) | وحدةتحكم |
| Conversions | تحويلات |
| Converting input to subscripts | تحويل مدخلات إلى دلائل |
| Counters | عدادات |
| Counter of active entries | عداد المحتويات النشيطة |
| Coupling | تقارن |
| CR (picture character) | CR ( رمز صورة ) |
| CRT | أنبوبة أشعة الكاثود |
| CRT (terminals) | نهايات طرفية بأنبوب أشعة الكاثود |
| Cylinder | اسطوانة |

# D

| | |
|---|---|
| Data Codes | شفرات ( رموز ) بيانات |
| Data division | جزء البيانات |
| Data Records | سجلات تاريخ ' |
| Date - Compiled | تاريخ الترجمة |
| Date fields | حقل تواريخ |
| Date - Written | تاريخ الكتابة |
| DAY | يوم |
| DB (picture character) | DP ( رمز صورة ) |
| Debug - Item | عنصر تصحيح |
| Debugging | تصحيح |
| Debugging Declarative | توضيحات تصحيح |
| Debugging lines | أسطر تصحيح |
| Decimal Point | علامة عشرية |
| Decision symbol | رمز قرار |
| Decision table | جدول قرارات |
| decoding input fields | فك شفرة حقول مدخلات |
| diagram | رسم |
| Decision trees | أشجار قرارات |
| DECLARATIVES | توضيحات |
| definition of | تعريف |
| Decoding input fields | فك شفرة حقول مدخلات |
| deleting an entry | حذف محتوى |
| Deletion code | رمز الحذف |

| | |
|---|---|
| DESCENDING | تنازلى |
| DESCENDING KEY | مفتاح تنازلى |
| Descending sort | ترتيب تنازلى |
| detailed desing | تصميم تفصيلى |
| Designing program logic (main loop) | تصميم منطقة البرنامج ( دورة رئيسية ) |
| Diagnostic error messages | رسائل أخطاء تشخيصية |
| Direct-access device | وحدة اتصال مباشر |
| Directory | دليل |
| Disk: | قرص |
| DISPLAY | أعرض |
| diagram | رسم |
| Disk drive | مشغل أقراص |
| Disk pack | مجموعة أقراص |
| Displacement | إزاحة |
| DISPLAY | اعرض |
| DISPLAY data | اعرض بيانات |
| DISPLAYUSAGE | الاستخدام للعرض |
| DIVIDE | اقسم |
| DIVISION | قسمة |
| Division beader | عنوان قسمة |
| Do while | افعل أثناء |
| Dollar sign | علامة دولار ( رمز تنقيح ) |
| DOWY BY | لأسفل بـ |
| Dump | نفايا |
| dumb | غبية |
| Duplication Factor | معامل ازدواج |
| Dynamic memory dump (DISPLAY) | نفايا ذاكرة ديناميكية ( اعرض ) |

## E

| | |
|---|---|
| EBCDIC | الشفرة الثنائية المؤسسة للتبادل البشرى |
| Edit program | برنامج تنقيح |
| Editing charaeter | رمز تنقيح |
| Editing | تنقيح |
| EJECT | يلفظ |
| Elementaryitem | عنصر فردى |
| ELSE | وإلا |

| | |
|---|---|
| ELSE NEXT SENTENCE | وإلا فالجملة التالية |
| End of file | نهاية الملف |
| ENVIRONMENT DIVSION | جزء الاوساط |
| EQUAL | يساوى |
| Errors: | أخطاء |
| I-O | مدخلات ومخرجات |
| Evalution | تقييم |
| EVALUATE | قوّم |
| Example | مثال |
| Exception conditions | شروط استثناءات |
| Execution of Stored program | تنفيذ برنامج مخزن |
| Executive module | جزء تنفيذ |
| EXHIBIT | عرض |
| External name | اسم خارجى |
| EXTEND | موسع |

# F

| | |
|---|---|
| Fatal logic | منطقى جسيم |
| Fatal logic errors | أخطاء منطقية جسيمة |
| FD entry | محتوى وصف الملف |
| Field | حقل |
| Figurative constant | ثابت استعارى |
| File | ملف |
| FILE-CONTROL | التحكم فى الملفات |
| File creation | إنتاج ملف |
| File description entry | محتوى وصف الملف |
| File labels | عناوين الملفات |
| File maintenance | صيانة الملف |
| File merging | دمج الملفات |
| File organization | تنظيم الملف |
| File retrival | استرجاع ملف |
| FILE SECTION | قسم الملفات |
| File sorting | ترتيب الملف |
| Fixed - length | ثابتة الطول |
| Fixed | ثابت |

| | |
|---|---|
| FILE STATUS | حالة الملف |
| File updating | تجديد الملف |
| FILLER | مالىء |
| First-record processing | تشغيل أول سجل |
| First-record switch | مفتاح أول سجل |
| Fixed head-per-track disk | رأس ثابت لكل مسار |
| Flags, | إشارات |
| Floppy disk | قرص مرن |
| Floating | متحركة |
| Flowchart | خريطة مسار |
| Footing area, | منطقة نهايات |
| For sort file | لملف الترتيب |
| From input data | من بيانات المدخلات |
| Functional cohesion | تماسك وظيفى |

## G

| | |
|---|---|
| Garbage | نفايا |
| General design | تصميم عام |
| GIVING | معطيا |
| GO TO | اذهب إلى |
| Grandparent, parent, child back | نسخ احتياطية للجد والأب |
| GREATER THAN | أكبر من |
| Group | مجموعة |
| Group item | مجموعة عناصر |
| Groupitem for key | مجموعة عناصر للمفتاح |
| Group MOVE | نقل جماعى |

## H

| | |
|---|---|
| Hard copy terminals | نهايات طرفية لإنتاج نسخ دائمة |
| Hard copy | نسخة دائمة |
| Hard disk | قرص ثابت |
| Hardware | نظم مكونات |
| Header | عنوان |
| Header label | عنوان امامى |
| Hierarchy chart | خريطة هرمية |

| | |
|---|---|
| HIGHVALUES | أعلى قيمة فى تسلسل التتابع |
| Higher-level language | لغة مرتفعة المستوى |
| HIPO chart | خريطة هيبو |
| Horizontal numbering | ترقيم أفقى |

# I

| | |
|---|---|
| I-0 | مدخلات ومخرجات |
| I-O errors | أخطاء مدخلات ومخرجات |
| IAR (Instruction Address Register) | مسجل عنوان إحدى التعليمات |
| IDENTIFICATION DIVISION | جزء التعريف |
| IF | إذا |
| indentation of nested | ترحيل التداخل |
| Input procedure ... Giving | أجزاء مدخلات ... معطيا |
| Input procedure.. output procedure | أجزاء مدخلات .. أجزاء مخرجات |
| Imperative statements | عبارات أمرية |
| Implicit redefinition | إعادة تعريف ضمنية |
| Indentation | ترحيل |
| Independent item | عنصر مستقل |
| Index | فهرس |
| Index records | سجلات فهرس |
| index required | مطلوب فهرس |
| INDEXED BY | مفهرس بواسطة |
| Indexed files | ملفات مفهرسة |
| indexing | فهرسة |
| Infinite loop | دورة لا نهائية |
| Initialization | وضع قيم ابتدائية |
| Input | مدخلات |
| Input devices | وحدات مدخلات |
| INPUT-OUTPUT SECTION | قسم المدخلات والمخرجات |
| Input-output symbol | رمز مدخلات ومخرجات |
| INPUT PROCEDURE | إجراء مدخلات |
| Input-process-output cycle | دورة مدخلات – عمليات – مخرجات |
| inserting new entry insources | ادخال محتوى جديد على التتابع |
| Insertion sort | ترتيب الإدخال |
| Insertion | إدخال |

| | |
|---|---|
| INSTALLATION | مؤسسة |
| Interblock gap | فراغ ما بين المجموعات |
| Intelligent | ذكية |
| Intermediate key | مفتاح متوسط |
| Invalid | غير صحيح |
| INTO | فى |
| IRG (interrecord gap) | فراغ ما بين السجلات |
| Iteration structure | هيكل التكرار |

## J

| | |
|---|---|
| Job control language (JCL) | لغة تحكم العمل |
| Julian date | تاريخ برقم اليوم |
| JUSTIFIED | مضبوط |

## K

| | |
|---|---|
| Key | مفتاح |
| Key field | حقل رئيسى ( مفتاح ) |
| Key-to-disk | من لوحة المفاتيح للقرص |
| Keypunch | التثقيب باللوحة |
| Kilobyte (KB) | كيلو بايت |

## L

| | |
|---|---|
| LABEL RECORDS | سجلات العنوان |
| Leading | رائدة |
| Length: | الطول |
| LESS THAN | اقل من |
| Level 01 | المستوى 01 |
| Level 77 | المستوى 77 |
| Level 88 | المستوى 88 |
| Level numbers | أرقام المستويات |
| LINAGE | خطى |
| Line image areas | منطقة صورة السطر |
| Linage | خطى |
| Linear IF | خطى |
| Linear table search | بحث جدول خطى |

| | |
|---|---|
| LINES AT BOTTOM | أسطر فى المؤخرة |
| LINES AT TOP | أسطر فى المقدمة |
| Literal: | ثابت |
| LOCK, WITH | اغلق بـ |
| Loading with read | تحميل بعبارة اقرأ |
| Logic error | خطأ منطقى |
| Loading | تحميل |
| Loading with value | تحميل بالقيمة |
| Logical deletion | حذف منطقى |
| Logical design | تصميم منطقى |
| Logical expressions, evaluation of | تعبيرات منطقية ، تقويمها |
| Logical operator | مؤثرات منطقية |
| Logical page | صفحة منطقية |
| Logical record | سجل منطقى |
| logical record copies | نسخ سجلات منطقية |
| Loop | دورة |
| LOW-VALUES | أقل قيمة فى تسلسل التتابع |

# M

| | |
|---|---|
| Machine language | لغة الآلة |
| Magnetic disk (see Disk) | قرص ممغنط |
| Magnetic tape | شريط ممغنط |
| Main program loop (logic design) | دورة برنامج رئيسية ( تصميم منطقى ) |
| Maintenance programming | برمجة صيانه |
| Major key | مفتاح رئيسى |
| Megabyte (MB) | ميجابايت |
| Memory dump | نفايا ذاكرة |
| MERGE, | ادمج |
| Merging | دمج |
| Merging unsorted files | دمج ملفات غير مرتبة |
| Microprocessor | ميكربروسسور – مشغل دقيق |
| Millisecond (ms) | ميللى ثانية |
| Minor key | مفتاح أصغر |
| Minus sign: | اشارةسالب |
| MIPS (millions instructions per second) | مليون من التعليمات فى الثانية |

| | |
|---|---|
| Misuse | استخدام خاطىء |
| MOVE | انقل |
| Moving variuble - length | نقل طول متغير |
| Multifile: | متعدد الملفات |
| Multikey sort | ترتيب متعدد المفاتيح |
| MULTIPLY | اضرب |
| Multivolume file | ملف متعدد الحجوم |

# N

| | |
|---|---|
| Name | اســـم |
| Nanosecond (ns) | نانو ثانية |
| NAMED | مسمى |
| NATIVE | تحديد تسلسل التتابع التقليدى |
| NEGATIVE | سالب |
| Nested PERFORM | نفذ متداخلة |
| NEXT SENTENCE | الجملة التالية |
| nested IF | إذا متداخلة |
| New master file | ملف رئيسى جديد |
| Nine (9, picture character) | 9 ( رمز صوره ) |
| Nonnumeric | غير عدد |
| Nonprinter | عددية |
| Nonfatal logic errors | أخطاء منطقية غير جسيمة |
| Numeric | غير طابع |
| NOT | ليس |
| Nonprinter | غير طابع |
| NUMERIC | عددى |
| Numeric data | بيانات عددية |

# O

| | |
|---|---|
| OBJECT-COMPUTER | كمبيوتر الهدف |
| Object program | برنامج الهدف |
| OCCURS | يحدث |
| Old master file | ملف رئيسى قديم |
| ON proced - name | على اسم اجراءات |
| ON | عند |

| | |
|---|---|
| ON All Procedures | على كل الأجراءات |
| ON All Prefmces | على كل الاشارات |
| ON File - name | على اسم الملف |
| ON - identifier | على معرف |
| On procedure - name | على اسم أجراءات |
| ON SIZE ERROR | عند حدوث خطأ فى الحجم |
| One - dimensional | ذو بعد واحد |
| OPEN | أفتح |
| OUPUT | مخرجات |
| Operation OF | عملية |
| Operating system | نظام تشغيل |
| Operational sign | اشارة تشغيل |
| Operator's console | نهاية طرفية لمشغل الكمبيوتر |
| OR | أو |
| ORGANIZATION IS | التنظيم يكون |
| Output | مخرجات |
| OUTPUT PROCEDURE | إجراء مخرجات |
| Overflow | سريان زائد |

# P

| | |
|---|---|
| P (editing character) | P ( رمز تنقيح ) |
| Packed-decimal data | بيانات عشرية مضغوطة |
| Padding | ملء فراغات أو أصفار |
| Page body | جسم الصفحة |
| Page overflow | سريان زائد للصفحة |
| Pagination without LINAGE | عمل صفحات بدون خطية |
| Paragraph | مقطع |
| Parallel tables | جداول متوازية |
| Parentheses (COMPUTE) | أقواس ( احسب ) |
| PERFORM | نفذ |
| Perform ... Varying | نفذ ... مغيراً |
| Period (editing character) | نقطة ( رمز تنقيح ) |
| Perioduse of, | نقطة ، استخدامها |
| Physical deletion | حذف واقعى |
| Physical record | سجل واقعى ( طبيعى ) |

| | |
|---|---|
| PIC (see PICTURE) | صورة |
| PICTURE | صورة |
| Picture characters | رموز الصورة |
| Plus sign: | اشارة موجب |
| POSITIVE | موجب |
| Predefined process symbol | رمز تشغيل سابق التعريف |
| Priming read | قراءة أولية |
| Printers | طابعات |
| Procedure Division Structure | هيكل جزء الأجراءات |
| PROCEDURE DIVISION | جزء الإجراءات |
| Process symbol | رمز عملية |
| PROGRAM COLLATING SEQUENCE | تسلسل تتابع البرنامج |
| Program constants | ثوابت البرنامج |
| Program development: | إعداد البرنامج |
| problem definition | تعريف المشكلة |
| PROGRAM-ID | تعريف البرنامج |
| Program logic | منطق البرنامج |
| Program stubs | اجزاء برنامج وهمية |
| Program testing | اختبار البرنامج |
| Program trace | تتبع البرنامج |
| Programmer-defined names | أسماء يعرفها المبرمج |
| Prompts | ملقنات |
| Pseudocode | شفرة شبيهة |
| Pseudcode equivalent | مكافىء الشفرة الشبيهة |

## Q

| | |
|---|---|
| Qualification | تأهيل |
| Quoted string | سلسلة موضوعة بين علامتى تنصيص |
| QUOTES | علامات تنصيص |

## R

| | |
|---|---|
| Random | عشوائى |
| Random processing | تشغيل عشوائى |
| READ | اقرأ |
| Read/write heads | رؤس قراءة وكتابه |

| | |
|---|---|
| Receiving field | حقل مستقبل |
| Record | سجل |
| Record codes | شفرات ( رموز ) سجل |
| RECORD CONTRAINS | السجل يحتوى |
| RECORD CONTAINS 0 CHARACTERS | السجل يحتوى صفر رمزاً |
| Record deletion | حذف سجل |
| Record description | وصف سجل |
| Record descriptor word | كلمة واصف سجل |
| Record insertion | إدخال سجل |
| Record names | أسماء سجلات |
| Record number | رقم سجل |
| Record updating | تجديد سجل |
| Recursion | إعادة ذاتيه |
| REDEFINES | يعيد تعريف |
| Reel | بكرة |
| Relation condition | شرط علاقى |
| Relative | نسبى |
| Relative indexing | فهرسة نسبية |
| RELEASE | يزيح |
| REMAINDER | الباقى |
| REMOVAL, CLOSE FOR | إزالة ، قريبا لـ |
| Repetition factor | معامل تكرار |
| Replacing relations | علاقات أحلال |
| RESERVE | محجوز |
| Reserved words, | كلماتمحجوزة |
| RESET TRACE | إعادة إعداد التتبع |
| RETURN | أعد |
| Retrieval time | وقت استرجاع |
| Reversed | عكس |
| REWIND , WITH NO | إعادة لف ، برقم |
| Restrictions on perform | قيود على نفذ |
| REWRITE | اعادة كتابة |
| Rotational delay | تأخير الدوران |

| | |
|---|---|
| ROUNDED | مقربا |
| Rounding | التقريب |
| Rule for nesting | قاعدة إعادة ذاتية |

# S

| | |
|---|---|
| S (picture character) | S ( رمز صورة ) |
| SD | وصف ملف الترتيب |
| SEARCH | ابحث |
| SEARCH ALL, | ابحث كل |
| SECTION | قسم |
| Section header | عنوان قسم |
| SECURITY | أمن |
| SELECT | اختر |
| Selction structure, | هيكل اختيار |
| Self-documenting language | لغة توثيق ذاتى |
| Sending field | حقل راسل |
| Sentences, | جمل |
| Sequence check | اختبار تتابع |
| Sequence structure | هيكل تتابع |
| Sequential | تتابعى |
| Sequential file updating | تجديد ملف تتابعى |
| Sequential processing | تشغيل تتابعى |
| Sequential table search | بحث جداول تتابعيا |
| SET: | ضع |
| Severe error | خطأ جسيم |
| SIGN | إشارة |
| Sign condition | شرط اشاره |
| Sign control editing | تنقيح تحكم الإشارة |
| Simple | بسيطة |
| Simple conditions | شروط بسيطة |
| SIZE ERROR | خطأ الحجم |
| SKIP 1, SKIP 2 SKEP 3 | اقفز ١، اقفز ٢ ، اقفز ٣ |
| Slack bytes | بايت راكدة |
| Salsh (editing character) | شرطة مائلة ( رمز تنقيح ) |
| SORT | رتب |

| | |
|---|---|
| Sort file | ملف ترتيب |
| Sort keys, | مفاتيح الترتيب |
| Sort/merge utility | تسهيلات الدمج والترتيب |
| SOURCE-COMPUTER | كمبيوتر المصدر |
| Source documents | وثائق المصدر |
| Source program | برنامج المصدر |
| SPACES | فراغات |
| Special characters | رموز خاصة |
| SPECIAL-NAMES | أسماء خاصة |
| STANDARD - 1 | نمطى – ١ |
| Statements | عبارات |
| STOP literal | قف ثابت |
| STOP RUN | أوقف التنفيذ |
| Strored PROGRAM, EXECUTION OF | برنامج مخزن ، تنفيذه |
| Structure | هيكل |
| Structure chart | خريطة هيكل |
| Structured flowchart | خريطة مسار مرتبة |
| Structured period | نقطة مرتبة |
| Stubs | أجزاء وهمية |
| Subfield | حقل جزئى |
| Subscriping | عمل أدلة |
| Subscripts | ادلة أو دلائل |
| Subscript conversion | تحويل الدليل |
| SUBTRACT | اطرح |
| Switches | مفاتيح |
| Symbolic dump | نفايا رمزية |
| SYNC (see SYNCHRONIZED) | متزامن |
| SYNCHRONIZED | متزامن |
| Syntax: | تكوين |
| Syntax error | خطأ تكوينى |
| Syntax notation | ترميز تكوينى |
| SYSIN | اسم خاص لوحدة مدخلات |
| SYSOUT | اسم خاص لوحده مخرجات |
| Systems flowchart | خريطة مسار النظام |

# T

| | |
|---|---|
| Table | جدول |
| two-dimensional summing entries | تجميع محتويات ذات بعدين |
| Table look-up: | فحص جدول |
| Table sort | ترتيب جدول |
| Subscriptconversion | تحويل دليل |
| Tape (magnetic) | شريط ( ممغنط ) |
| Terminal: | نهاية طرفية |
| Terminal Symbol | رمز نهاية طرفية |
| Test data | بيانات اختبارية |
| Testing | اختبار |
| THRUOPTION | جزء خلال |
| TIME | وقت |
| TIMES | مرة إلى |
| THRU TO | خلال |
| Top-down, modular design | تصميم إجزاء من القمة إلى القاعدة |
| Top margin | هامش علوى |
| To | لـ |
| Trace | تتبع |
| Trailer label | عنوان خلفى |
| Trailing | متأخرة |
| Transaction file | ملف عمليات جارية |
| Transaction register | مسجل عمليات جارية |
| Truncation | حذف |
| Tow level | مستويان |
| Two-dimensional tables | جداول ذات بعدين |
| Typing rules | قواعد الكتابة |
| Top - down coding and design | كتابة شفرة وأختبار من القمة الى القاعدة |

# U

| | |
|---|---|
| Unary minus | سالب أحادى |
| Undefined, | غير معرف |
| Unit | وحده |
| Unit record devices | وحدات سجل وحدة |

| | |
|---|---|
| Until in cobol | حتى فى كوبل |
| Until | حتى |
| Unsorted files | ملفات غير مرتبة |
| Update in place | تجديد فى نفس المكان |
| Up By.. | لأعلى بـ .. |
| USAGE | استخدام |
| Use in if | مستخدم فى إذا |
| Use in perform | مستخدم فى نفذ |
| USE FOR DEBUGGING | يستخدم فى التصحيح |
| USING | مستخدما |
| Utility programs | برامج تسهيلات |

# V

| | |
|---|---|
| V (picture character) | V ( رمز صورة ) |
| Validation | التأكد من الصحة |
| VALUE | قيمة |
| versus MOVE | عكس نقل |
| Variable length | متغيرة الطول |
| Variable-length records | سجلات متغيرة الطول |
| Variable-length tables | جداول متغيرة الطول |
| Varying | مع تغيير |
| Verbs | أفعال |
| Versus Value | ضد القيمة |
| Vertical numbering | ترقيم رأسى |
| Virtual storage | تخزين افتراضى |
| Visual table of contents | جدول مرئى للمحتويات |
| Volume | حجم |
| Volume label | عنوان حجم |
| Volume switching | تغيير الحجم |
| Volume table of contents (VTOC) | حجم جدول المحتويات |

# W

| | |
|---|---|
| Warning | تحذير |
| WHEN | عندما |
| WITH DEBUGGING MODE | مع حالة التصحيح |

| | |
|---|---|
| WORKING-STORAGE | مخزن العمل |
| WRITE | اكتب |

## X

| | |
|---|---|
| X (picture character) | X ( رمز صورة ) |

## Z

| | |
|---|---|
| Z (picture character) | Z ( رمز صورة ) |
| ZERO | صفر |
| Zero (editing character) | صفر ( رمز صورة ) |

دار المعارف

# PROGRAMMING WITH STRUCTURED COBOL ( Schaum )

## صدر أيضاً للناشر
## فى
## الحاسبات

* البرمجة بلغة الباسكال ........................................ كيلر
* الدوائر المتكاملة الرقمية والحاسبات .............................. وولارد
* المجهزات والحاسبات الدقيقة لطلبة الهندسة والفنيين .................. وولارد
* الدوائر المنطقية واستخدامات المجهزات الدقيقة ...................... موريس
* المدخل لعلم الحاسبات ............................................ بارتي
* البرمجة بلغة الباسكال ( شوم ) ................................... جوتفريد
* البرمجة بلغة البيسك ( شوم ) ..................................... جوتفريد
* البرمجة بلغة الفورتران ( شوم ) .................................. ليبشتز
* الرياضيات الاساسية للحاسب ( شوم ) ............................... ليبشتز
* كويك بيسك ...................................................... ناميروف
* موسوعة مصطلحات الكمبيوتر ........................................ محمود الشريف


الناشر

الدار الدولية للنشر والتوزيع

٣٨ ش الاهرام – روكسى – مصر الجديدة
ص.ب : ٥٥٩٩ هليوبوليس غرب – القاهرة
ت : ٢٥٨٢٨٨٧
تلكس : ٢٠٠٧٠ PBCRB UN
فاكس : ٢٩١٨٠٥٩ / ٢٠٢٠٠

ISBN : 977 - 5107 - 26 - 1